Computer Networks 103 (2016) 101–114

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

TCP over scarce transmission opportunity in cognitive radio networks

Sang-Seon Byun*

Department of Computer Engineering, Catholic University of Pusan, Busan, South Korea

ARTICLE INFO

Article history:
Received 6 July 2015
Revised 29 January 2016
Accepted 20 March 2016
Available online 6 April 2016

Keywords:
TCP
Freeze-TCP
TCP-Freeze-CR
Cognitive radio networks
Software-defined radio
Universal Software Radio Peripheral
GNU Radio
IEEE 802.15.4

ABSTRACT

Transmission control protocol (TCP) is the most popular transport layer protocol for applications that require reliable and ordered data delivery essentially. In this paper we consider the deployment of TCP to secondary users (SUs) in overlay cognitive radio networks (CRNs), and address its performance degradation; in CRNs, SU's transmissions are frequently disrupted by the detection of primary user's transmission, and which makes the SU experience consecutive retransmission-timeout and its exponential backoff. Subsequently, the TCP in SU does not proceed with the transmission even after the disruption is over or the SU hands over to other idle spectrum. To tackle this problem, we propose a cross-layer approach called TCP-Freeze-CR; lower layer protocols send the overlying TCP two different cross-layer signals, *freeze* on the detection of primary user's transmission, and *unfreeze* after handing over to an idle spectrum. Moreover we consider a practical situation where either secondary transmitter (ST) or secondary receiver (SR) detects primary user's transmission; therefore additional message exchanges are needed between ST and SR to retrieve and resynchronize to other idle spectrum, i.e., spectrum synchronization. This situation is more complex than the case where both ST and SR detect primary user's transmission. Hereby, we develop a spectrum synchronization procedure coupled with TCP-Freeze-CR into a finite state machine. All of our proposals are implemented and evaluated on a real CRN consisting of 6 software radio platforms. In the implementation, we deploy 802.15.4 implementation as a target physical layer protocol, and couple it with TCP-Freeze-CR using Unix Domain Socket. The experimental results illustrate that standard TCP suffers from significant performance degradation in CRNs, and show that TCP-Freeze-CR can greatly alleviate the degradation; e.g., for 1200 s, ST with TCP-Freeze-CR can send about 10 times more packets than ST with standard TCP.

1. Introduction

Cognitive radio (CR) technology plays an essential role for solving the problem of spectrum scarcity as demand for emerging wireless applications is increasing: smart power grid, medical appliances, surveillance system, etc. To this end, a plentiful number of research literature have been published since Mitola and Maguire proposed the concept of CR [1]. However, majority of the research work related to CR have focused on the issues in physical or link layer, i.e., spectrum sensing, dynamic spectrum allocation, interference control, etc.; a few of them have addressed issues in routing over CR ad-hoc networks.

In this paper, we focus on significant performance degradation that transmission control protocol (TCP) suffers from in CR networks (CRNs), and study an enhancement scheme that can alleviate the performance degradation.

In Internet, TCP provides applications with reliable and ordered packet delivery over unreliable physical media. Therefore, TCP is the most widely used transport layer protocol even in wireless networks, and there is no signs of change in the foreseeable future.

In TCP-NewReno [2],[1] packet losses are regarded as the signal of network congestion, and two different signals are used for noticing the packet loss: triple duplicate acknowledgments (ACKs) and retransmission-timeout (RTO). On receiving triple duplicate ACKs, TCP performs fast retransmission and triggers congestion avoidance mechanism with halving its congestion window size (henceforth, we denote congestion window size *cwnd*); on the occurrence of an RTO, it retransmits unacknowledged packet with squeezing *cwnd* to 1 and triggers the slow start deeming the network overloaded.

An extensive research work has been conducted to resolve the problem that standard TCP experiences over conventional wireless networks: multiple random packet losses within one round-trip-time (RTT) due to interference, shadowing, fading, and collision in

* Tel.: +82 51 510 0651.
E-mail address: ssbyun@cup.ac.kr

[1] Throughout this paper, TCP-NewReno is referred as standard TCP since it has been the most widely used TCP standard [3].

http://dx.doi.org/10.1016/j.comnet.2016.03.026
1389-1286/

wireless channels, lead to consecutive RTOs and exponential back-off of retransmission-timer (RT), and standard TCP therefore suffers from drastic decrease of its throughput [4].

In CRNs, a secondary user (SU) whose applications communicate using standard TCP experiences performance decrease because of not only but the random packet loss and interference also disruption by primary user (PU) 's transmission (hereafter, referred as primary transmission briefly) especially if both of the SU and PU are accessing the same channel [5]. Generally, the random loss occurs transiently, but on the other hand, the disruption by primary transmission may last for a relatively long while. Therefore, unless SU hands over to other unused spectrum, the disruption can incur more consecutive RTOs and exponential back-off of RT; assuming such disruption triggers an RTO at an SU, and continues even until the next RT expires, the RT is backed off exponentially twice, and thus the SU does not proceed with the transmission even after the disruption is over or she succeeds to hand over to other unused spectrum. This performance degradation becomes more significant as primary transmission occurs more frequently and for a longer while.

Most of the TCPs for conventional wireless networks also tackle the same issue: discerning congestive error and channel error. However the channel error in conventional wireless networks is generally transient, and most of the TCP for conventional wireless networks control sending rate or window size with reflecting the channel error to bandwidth estimation. However SU should shut down the transmission completely as soon as detecting primary transmission (henceforth, we note primary user detection). This feature is the difference with TCPs for conventional wireless networks.

Considering an overlay-CRN[2]—whose definition is given in later section of this paper—with multiple channels, we envisage a cross-layer approach that can alleviate the aforementioned problem with modifying standard TCP very slightly. The modified TCP is referred to as *TCP-Freeze-CR*. TCP-Freeze-CR is implemented in secondary transmitter (ST) only; secondary receiver (SR) can use the standard TCP without any modifications.

As inferred in the name of the scheme, TCP-Freeze-CR is motivated by Freeze-TCP [6]. In Freeze-TCP, wireless receiver predicts impending channel degradation and mobility, and freezes its TCP via cross-layer signaling. On the prediction, wireless receiver sends zero window advertisement in order to freeze the transmission. On the detection of good channel quality, the wireless receiver sends triple ACKs then its sender resumes the transmission. As mentioned beforehand, TCP in overlay-CRNs should shut down transmission on primary user detection quickly. Therefor Freeze-TCP is the best candidate that can be applied to overlay-CRNs easily.

Besides, we consider a realistic situation where spectrum handover is triggered by either ST or SR. Therefore, any SU detecting primary transmission should notify the other SU of the detection since both ST and SR should be made perform spectrum sensing to retrieve an idle spectrum commonly accessible by themselves. To this end, we develop a spectrum synchronization mechanism using a finite state machine, and couple it with TCP-Freeze-CR.

We evaluate our scheme by implementing it onto a representative software-defined radio (SDR) platform, Universal Software Radio Peripheral (USRP E100), where many part of the lower layer operations are executed as user processes written in high level programming languages, i.e., C++ and Python, above the library called USRP Hardware Driver (UHD) [7]. Furthermore, we deploy IEEE 802.15.4 implementation [8] as the target lower layer protocols, and use orthogonal frequency-division multiplexing (OFDM) implementation for PUs.

The remainder of the paper proceeds as follows: In Section 2, we present our spectrum synchronization mechanism. In Section 3, we present our main contribution, TCP-Freeze-CR. In Section 4, we give the implementation details. In Section 5, we present the experimental results. In Section 6, we give some related work. In Section 7 we discuss an extension of our scheme to multi-hop CR networks, and conclude this paper in Section 8.

2. Spectrum synchronization

2.1. Access technique for CRNs

Access techniques for CRNs can be classified into two types [5]. In overlay-CRNs (or interference-free CRNs), SUs can access spectrums not occupied by any PUs. As a result, there should be virtually no interference to the PUs. On the other hand, in underlay-CRNs (or interference-tolerant CRNs), SUs are allowed to interfere with primary transmission up to a certain tolerable level. In this paper, we consider the overlay-CRNs only.

2.2. Spectrum sensing model

Generally, spectrum sensing is defined as the task of finding spectrum unused or underutilized by PUs [9]. Then spectrum sensing approaches can be classified into two types according to the time when the sensing task is performed: *proactive sensing* and *on-demand sensing*. When proactive sensing is applied, SUs perform the sensing task periodically and continuously even while they are communicating over an idle channel safely; on detecting primary transmission, ST and SR hand over and resynchronize to the spectrum detected as idle by both of them. On the contrary, SUs perform the sensing task only when they overhear primary transmission in on-demand sensing.

Assuming that ST finds the spectrum to hand over and resynchronize, ST should receive the list of the spectrums detected as idle by her target receiver (i.e., SR) as well. Then the ST should inform the SR of the spectrum they will hand over and resynchronize to. Therefore additional message exchanges between ST and SR are inevitable. When proactive sensing is used, the message exchanges occur periodically and continuously, and thus, proactive sensing can prevent PUs from overhearing these message exchanges. However, on-demand sensing cannot eliminate such overhearing (unless there is an extra dedicated channel for the message exchange).

In this work, we consider the spectrum synchronization coupled with on-demand sensing. Hence we need to assume an extra control channel for the message exchanges between ST and SR. By applying proactive sensing, we can relax this assumption. However we have found that proactive sensing is not adequate on our software radio platform (i.e., USRP E100) since periodic and continuous spectrum sensing overuse the processing power. Thus we decide to use on-demand sensing unavoidably, and, as a result, more processing resources are given to processing of TCP-Freeze-CR and 802.15.4 implementation. Incidentally many related work such as [10] and [11] have premised ideal proactive sensing without any implementation-based experiments.

2.3. Spectrum synchronization model

Majority of related work have assumed that ST or a central authority can detect all primary transmissions or idle spectrums that are actually detectable by SR only [10]. In this paper, we relax this assumption; that is, we consider the situation where ST and

[2] In this paper we consider only single-hop networks. Nonetheless, we believe that our approach can be extended for multi-hop or infra-structured networks, and which remains as one of our future task.