MSHCS-MAC: A MAC protocol for Multi-hop cognitive radio networks based on Slow Hopping and Cooperative Sensing approach

Nhat Pham, Kiwoong Kwon, Daeyoung Kim
School of Computing
Korea Advanced Institute of Science and Technology,
Daejeon, Republic of Korea
Emails: {nhatphd, kiwoong, kimd}@kaist.ac.kr

***Abstract*—Since the concept of Cognitive Radio (CR) was first introduced, it has been considered as a key technology of future wireless devices to better utilize radio spectrum. A number of CR-MAC protocols have been studied for Cognitive Radio Networks (CRNs), but most state-of-the-art frequency-hopping based protocols focus mainly on channel mobility and spectrum resource allocation. They lack integration of other essential features such as time synchronization and cooperative spectrum sensing, which are practically crucial, into a full-blown CR-MAC protocol. In this paper, we propose MSHCS-MAC: a mac protocol for Multi-hop CRNs based on Slow Hopping and Cooperative Sensing approach which is the cutting edge frequency hopping scheme. The contributions of MSHCS-MAC are threefold: (1) Support of multi-hop communication without dedicated control channel and multiple transceivers, (2) Integration of essential CR-MAC features such as bootstrapping, multi-channel operation, cooperative spectrum sensing and time synchronization, (3) Practical implementation and evaluation on commercial devices. The evaluation results show that MSHCS-MAC provides reasonable performance in the experimental testbed with supporting multi-hop communication and essential CR-MAC features.**

***Keywords*-Cognitive radio networks, software defined radio, medium access control (MAC), slow hopping, cooperative sensing, opportunistic spectrum access.**

I. Introduction

Since the idea of Cognitive Radio was first introduced, it has been gaining lots of attention from both academia and industry community as the demand for wireless spectrum utilization increases rapidly in nowadays applications. On the other hand, measurements in several reports [1], [2] show that our traditional static spectrum allocation scheme is inefficient. It prevents rarely used frequencies from being used by other users resulting in spectrum wastage. CR technology leverages spectrum wastage problem by enabling wireless devices, called secondary users (SUs), to sense wireless channels and perform opportunistic access to spectrum white spaces while evacuating the channel as soon as primary users (PUs), who have a license to use that channel, appear. This allows us to utilize spectrum resource more efficiently. In recent years, studies for CR in both academia and industry have been focusing on introducing cognitive access in licensed television spectrum bands, creating international wireless standards for supporting secondary access of licensed spectrum, and incorporating CR technology into existing standards.

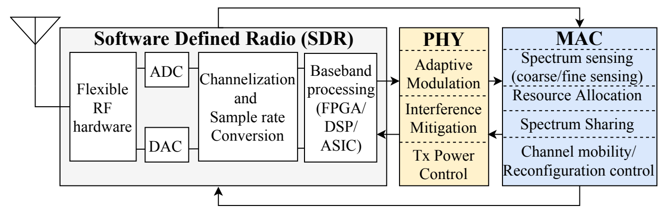


Figure 1. A general Cognitive Radio MAC architecture.

CR-MAC protocol has an important role in several cognitive functions, as presented in Fig. 1, such as spectrum sensing, channel mobility, resource allocation, and spectrum sharing. Spectrum sensing is the ability to gather information about surrounding wireless environment in order to keep a dynamic picture of available channels. Channel mobility helps CR devices to communicate over many channels and evacuate to a different channel as soon as licensed users appear. Resource allocation is exploited to opportunistically assign available channels to CR devices. Spectrum sharing allows CR devices to avoid harmful contentions and interferences with PUs.

The challenges of CR-MAC protocols lie in its opportunistic nature. While the number of available channels is fixed in classical multi-channel networks, it varies with time and space in CR networks. The requirement of avoiding interference with PUs creates a dynamic available channels map over time. Opportunistic access to different channels in various time periods also leads to the multi-channel hidden terminal problem. Furthermore, without previous knowledge about PU activity, SU has no idea when PU will appear on its working channel. As a result, CR device has to exploit sophisticated sensing method to periodically detect new white spaces as well as to protect primary transmissions.

There are numerous CR-MAC protocols which have been proposed in the literature. Each of them has both pros and cons. While common control channel based protocols suffer from control channel saturation and jamming vulnerability, communications in frequency-hopping based protocols are more reliable as they do not rely on a dedicated channel.

Previous works, following frequency hopping approach such as Synchronized MAC (SYN-MAC) [3] and Dynamic channel Hopping MAC (DH-MAC) [4], focus mainly on channel mobility and resource allocation with assumptions that spectrum sensing and time synchronization functions have been provided. Since spectrum sensing and time synchronization have essential roles in preventing interferences with PU activities and synchronizing among SUs in frequency hopping scheme, lacking the integration of them creates a challenge for previous protocols to be employed in practical networks. Incorporating these functions in multihop CR network is not a trivial task as we face challenges of contentions, unsynchronized quiet sensing period, and synchronization errors accumulation. Thus, their evaluation results are mostly from ideal environments of simulations. Moreover, they either have much overhead in term of hardware or transmission efficiency.

In our previous work, a state of the art Slow Hopping based Cooperative Sensing MAC protocol (SHCS-MAC) has been proposed in [5]. It was the first protocol which includes a cooperative sensing function in the channel rendezvous and common hopping approaches. Although SHCS-MAC has many advantages in improving aggregate throughput, better coexisting with PUs, SUs and minimizing radio cost (i.e. only one transceiver), the proposed protocol only works within one hop. Besides, it was evaluated only in a simulation software. As a result, practical applications of SHCS-MAC is limited.

In this paper, we propose MSHCS-MAC: A mac protocol for multi-hop CRNs based on Slow Hopping and Cooperative Sensing approach which is the cutting edge frequency hopping scheme. The contributions of MSHCS-MAC are threefold: (1) it is able to perform cognitive communication over multiple hops while eliminating the needs of dedicated control channel and only one radio is required, (2) it provides an integration of essential CR-MAC features such as bootstrapping, multi-channel operation, cooperative spectrum sensing, and time synchronization into a fully functional CR-MAC protocol, (3) the protocol was implemented on real devices (i.e. Linux PCs and Android mobile devices) by using GNU Radio and USRPs, and evaluated under practical network deployment. The evaluation results show that MSHCS-MAC provides reasonable performance in the experimental testbed with supporting multi-hop communication and essential CR-MAC features.

The rest of the paper is organized as follows: Sec. II provides a review of related works in this area. Our proposed protocol is introduced in Sec. III. Its implementation and evaluation results are presented in Sec. IV. Finally, we will conclude our paper in Sec. V.

II. Related Works

A. De Domenico et al. provide a comprehensive survey for state of the art CR-MAC protocols in [6]. Based on the strategy to exchange control information, we can divide CR-MAC protocols into three groups: (1) common control channel (out-of-band), (2) split phase (out-of-band) and (3) frequency hopping (in-band).

Cross-layer based opportunistic MAC [7], and Opportunistic Spectrum MAC (OS-MAC) [8] are protocols which fall into the first category. These protocols use a dedicated control channel to exchange signaling information, sensing outcome and channel selection. A channel, which is free from PU activity, in the unlicensed band could be used as the control channel. This scheme does not require strict synchronization of SUs to operate; however, a dedicated transceiver of each device must always stay in common control channel to avoid missing control signals. Furthermore, the common control channel is also prone to jamming and saturation.

Cognitive MAC (C-MAC) [9] and distributed Multichannel MAC protocol for CR networks (MMAC-CR) [10] are two examples of split phase scheme. In this scheme, only one transceiver can be used for both control signals and data transmission. Time synchronization technique is required to split time frames into control and data phases. In control phase, every CR nodes will rendezvous at the same channel to exchange control signals. Thereafter, data transmission will be performed concurrently at different channels. This scheme suffers from wasting free data channels during control phase and also from the use of a dedicated out-of-band channel.

While out-of-band control channel is exploited in common control channel and split phase schemes, in-band signaling is proposed in frequency hopping scheme. In this scheme, both control and data exchanges are performed in the same channel of licensed spectrum. Each CR devices will follow a hopping sequence and continuously hop to different channels until they are involved in data transmission. Data communication is more reliable as it does not rely on a single common channel. Furthermore, the use of only one radio could be achieved in this approach. However, frequency hopping scheme requires tight synchronization among network nodes to operate efficiently. SYN-MAC [3], DH-MAC [4] and SHCS-MAC [5] are several examples that follow this approach.

In [3], SYN-MAC is proposed for multi-hop CR networks where each device has two transceivers. While SYN-MAC can avoid using a dedicated control channel to exchange control messages, it requires each node to exchange available channels with other neighbors. Moreover, the use of two radios also increases the cost. In [4], DH-MAC presents a dynamic channel hopping approach which can outperform SYN-MAC with only one radio is needed. However, the overhead in this protocol is high as each node must frequently advertise its hopping sequences and monitor its neighbors' sequences. A sender in DH-MAC needs to follow its receiver's hopping sequence to perform data transmission. While time synchronization and cooperative spectrum

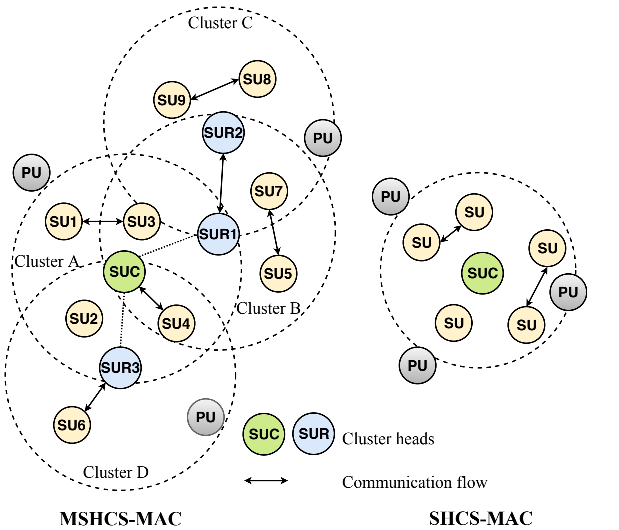


Figure 2. MSHCS-MAC network model (left) with 4 clusters (i.e A-D) and the original SHCS-MAC network model (right).

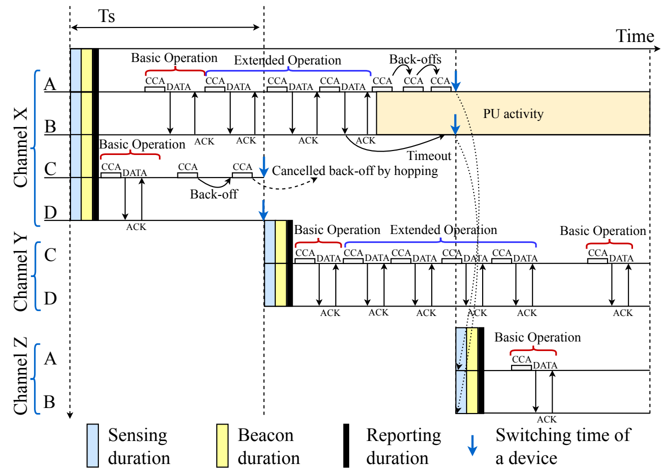


Figure 3. Data transmissions in MSHCS-MAC with basic and extended operations.

sensing are practically critical for frequency hopping based CR-MAC protocols, in previous works, these functions have not been fully discussed and integrated. The integration of spectrum sensing also creates challenges for time synchronization in CR-MAC protocols to prevent sensing and data durations from overlapping and causing false alarms in multi-hop scenarios.

III. Proposed MSHCS-MAC Protocol

A. System Model

Our proposed protocol is based on coordinator-based CR network and cluster tree topology with four different device types: primary user (PU), secondary user coordinator (SUC), secondary user relay node (SUR) and secondary user (SU). Fig. 2 shows the network models of MSHCS-MAC with four clusters and the original SHCS-MAC. PU is the one which has the license to use the spectrum while SUC, SUR, and SU are CR devices which perform opportunistic access to the spectrum. Each SUC/SUR with its associated SUs form a cluster with that SUC/SUR as the cluster head. We call this cluster is the local cluster of a SUC/SUR. SUR will connect to another cluster which is called parent cluster in our tree topology. Communication within a cluster could be established between two SUs or a SU with its cluster head. On the other hand, inter-cluster communication between two devices in different clusters always need to be forwarded to their corresponding cluster heads. We assume that only cluster heads have routing ability in our model.

MSHCS-MAC maintains frequency hopping scheme proposed in the original SHCS-MAC where time is divided into time slots (Ts). At the end of each time slot, every CR devices in our network will move to another common channel which is chosen at random. The sequence of selected channels is called common hopping sequence (CHS). Each time slot is further divided into sensing (Tss), beacon (Tb), reporting (Tr) and data duration ($Td = Ts - Tss - Tb - Tr$).

Active beacon broadcasting at each hop from cluster head is employed in our protocol to advertise network information such as network and cluster identifiers, durations for sensing, beaconing, reporting and data transmission, random seed to generate CHS, and a reference time stamp. This information is used for bootstrapping process when a node joins the network.

Data transmissions among CR devices are performed by contention-based CSMA/CA protocol. Transmissions could be performed in two ways: basic and extended operations (Fig. 3). Basic operations are used when the amount of data is small and it can fit within the data duration of a time slot. To improve aggregated throughput, slow hopping based multi-channel operation mechanism is used for extended transfer. In this case, when a sender-receiver pair has more data and needs to reserve the current channel for a longer time, both of them can stay in that channel without switching until their data transfer is finished or interrupted by PU activity. The sender can notify its receiver for extended operation by setting a flag in the packet header. After data transmission has been finished, it can return to the CHS to rendezvous with other devices. the receiver, on the other hand, waits for a timeout to expire before returning to CHS. To keep track of CHS, each CR device has to update its hopping sequence every time slot during extended operation.

B. Secondary User Coordinator and Secondary User nodes

SUC is the root of our tree topology. Its bootstrapping process begins with finding a channel to form a new CR network. Thereafter, it will start performing channel hopping and broadcasting beacon during beacon duration of each time slot. SU node bootstraps by listening for beacons in a channel. If it does not receive any beacon after a timeout has expired, it will move to another channel and continue listening until it receives a beacon. After a beacon has been

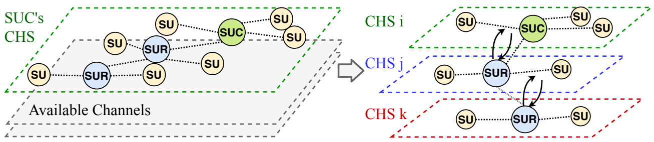


Figure 4. Following the same common hopping sequence (CHS) (left) over multiple hops is inefficient. Thus, each cluster is designed to have an independent CHS (right) to better utilize available spectrum.

received successfully, SU uses the information obtained from beacon to follow CHS and synchronize its time with its cluster head (SUC or SUR).

We assume that every node in our network use the same linear congruential random generator where a generated value is used as the seed to calculate next values in the sequence. Channel index is obtained by taking the modulo of the generated value and the total number of channel. That value is included in each beacon payload from SUC/SUR.

C. Secondary User Relay Node

To achieve multi-hop communication, new CR devices called Secondary user relay nodes (SURs) are added in our network model. Basically, for any two devices in our network to communicate with each other, they must be on the same channel or follow the same common frequency hopping sequence. However, as the network expands more contentions will happen if they follow the same CHS. Furthermore, time synchronization errors are accumulated over multiple hops which creates overlapping among devices' sensing and data durations resulting in false alarms. Moreover, following the same CHS is inefficient as it leaves other available unused. Hence, to effectively utilize multichannel environment and reduce contentions among CR devices, each cluster is designed to have independent CHS as shown in Fig. 4. Having independent CHSs is also a simple and elegant way to ensure fairness in spectrum access among clusters as each cluster will move to a random channel at each time slot.

Each SUR needs to follow two CHSs to keep the connection between its local and parent clusters. As the minimal radio cost (i.e. only one transceiver) is desired, SUR is designed to repeatedly switch between two CHSs by staying one time slot in its local cluster and another time slot in parent cluster. When SUR moves to its parent cluster, it will act as a normal SU while it performs similarly to a SUC in its local cluster.

Following two CHSs with only one radio also requires SUR's bootstrapping process to be performed in two stages. In the first stage, SUR listens for beacons from a cluster to synchronize with its parent cluster. Then, in the next time slot, SUR will establish its own cluster with an independent CHS and start broadcasting beacon.

While intra-cluster communications among CR devices in a cluster are trivial as they follow the same CHS, inter-cluster communications which involve SURs are more

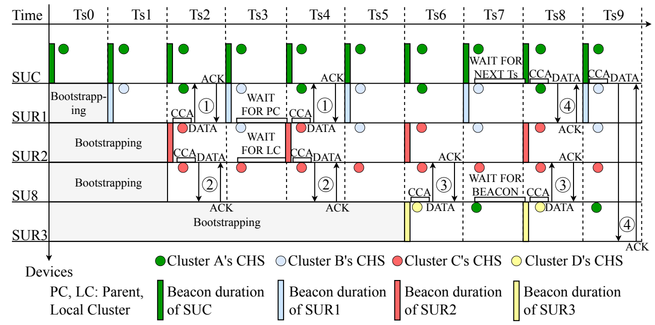


Figure 5. Handling *deafness problem* in communications which involve SURs.

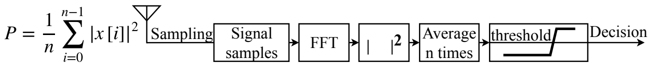
complicated. Since SUR does not always stay in a CHS, it creates so-called *deafness problem* mentioned in [11]. *Deafness problem* happens when two CR nodes can not communicate with each other because one of them (i.e. SUR in our case) have already moved to another channel. As a result, packets will be lost and our performance will degrade dramatically. This issue is handled in four cases which are illustrated in Fig. 5:

1) Transmission from SUR to its parent cluster head: SUR will need to wait until it switches to CHS of parent cluster to perform transmission. (E.g. SUR1 $\rightarrow$ SUC)
2) Transmission from SUR to SU in its local cluster: similarly, SUR needs to wait until it switches to the local cluster. (E.g. SUR2 $\rightarrow$ SU8)
3) Transmission from SU to its cluster head: SU needs to wait for beacons from SUR make sure that SUR has returned to the CHS. (E.g. SU8 $\rightarrow$ SUR2)
4) Transmission from SUC to SUR in its cluster: SUC needs to keep track the channel hopping index when a SUR associates with it. As SUR stays one time slot in local cluster and another time slot in parent cluster, it will rendezvous with SUC every two time slots. (E.g. SUC $\rightarrow$ SUR1 and SUR3)

D. Cooperative Spectrum Sensing

Spectrum sensing is a crucial function of CR-MAC providing the ability to detect PU activity and avoid harmful interferences. Since individual sensing is not sufficient to handle PU hidden problem due to shadowing [12] and multipath fading [13], it is necessary to employ cooperative sensing where all CR nodes cooperate together. The whole cooperative spectrum sensing procedure is performed in two phases: local sensing and reporting.

Local sensing is performed individually at each node during sensing period (Tss) which is a short quiet period lasting for several milliseconds at the start of each time slot. Synchronization among nodes is important here so no data transmission could interfere and create false alarms (i.e. falsely conclude the current channel is busy while



$$P = \frac{1}{n}\sum_{i=0}^{n-1} |x[i]|^2$$

Figure 6. Schematic representation of energy detection technique.

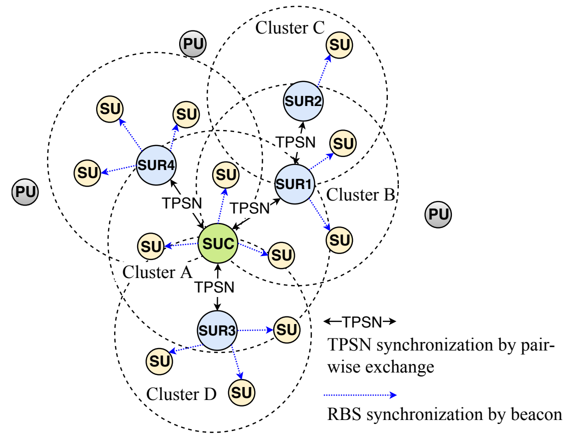
there is no PU activity). There are many techniques could be used for local sensing such as energy detection (ED), waveform-based detection, cyclostationarity, radio identification, matched filter. ED is employed in our design and implementation for the sake of simplicity. Other spectrum sensing techniques could also be used. ED is performed by measuring the average power of a wireless spectrum and compare it with a threshold which depends on noise floor of the spectrum. Schematic representation of ED technique is shown in Fig. 6. After sampling process, discrete wireless samples will be transformed to the frequency domain with FFT (Fast Fourier Transform) block. Then, measured power is calculated by taking the average of transformed samples' squared magnitudes.

After local sensing has been performed during sensing duration, classical OR-rule is employed in cooperative reporting in reporting duration (Tr). If a node successfully detects PU activity, it will broadcast busy signal for other devices in its range. CR nodes detecting busy signal will suspend any data transmission in the current time slot to prevent interference with PU. To reduce interferences with PU, Tr should be designed as short as possible. Thus, pilot signals could be used as busy signals as we do not need to include any data.

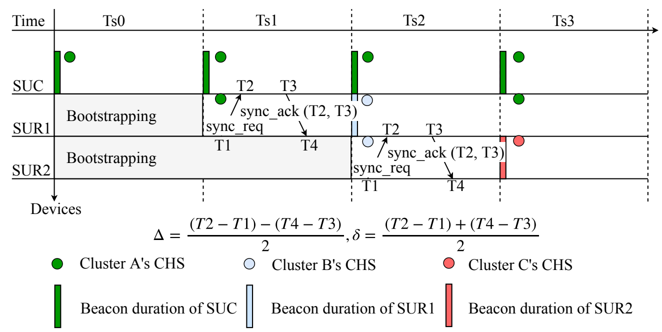
Sensing and reporting durations, on one hand, should be short comparing to the total duration of a time slot to reduce overhead on data transmission. On the other hand, these durations also depend heavily on the performance of time synchronization protocol. As a rule of thumb, they should be at least double the maximum offsets which can be achieved by time synchronization protocol to compensate for the overlapping sensing and data duration between two neighbor nodes. In other words, time slot duration and time synchronization protocol accuracy are upper and lower bounds, respectively, for both sensing and reporting durations.

E. Time Synchronization

At the cluster level, time synchronization is important so every node can perform cooperative sensing and data transmission together. Unsynchronized nodes can create false alarms during sensing period (i.e. overlapped sensing and data durations) and mismatches in their transmission as one node may have already moved to another channel while others are still trying to transmit to that node. Frequent mismatches in data transmissions will eventually lead to reduced packet delivery ratio among CR nodes. At the network level, time synchronization errors are accumulated after each hops creating scalability issue for multihop scenarios. Although each cluster is designed to have independent CHS, there are still chances which two or more clusters will stay on



(a) Hybrid synchronization scheme with TPSN and RBS.



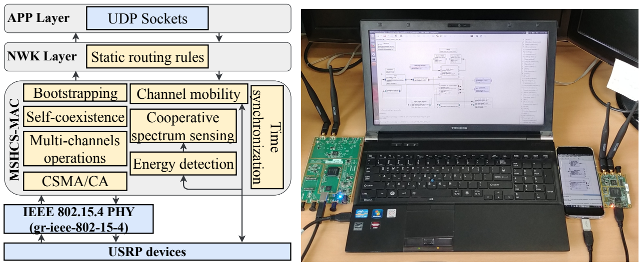
$$\Delta = \frac{(T2 - T1) - (T4 - T3)}{2}, \delta = \frac{(T2 - T1) + (T4 - T3)}{2}$$

(b) TPSN synchronization process of SUR1 and SUR2 after successfully associated with SUC and SUR1, respectively.

Figure 7. Time synchronization of MSHCS-MAC.

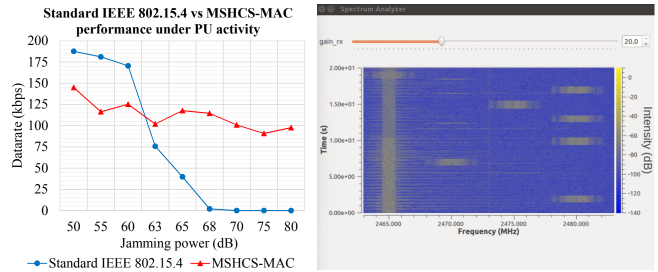
the same channel at the same time. The problem is worse when clusters are close to each other or when the number of clusters is more than the number of available channels. As a result, while a coarse-grained (i.e. milliseconds level) synchronization protocol can be used for devices within a cluster, a fine-grained one (i.e. microseconds level) should be used to synchronize clusters together.

There are several classical state-of-the-art time synchronization protocols in literature such as: Reference Broadcast Synchronization (RBS) [14], Timing-sync Protocol for Sensor Networks (TPSN) [15] and Flooding Time Synchronization Protocol (FTSP) [16]. Although FTSP is considered as the de-facto time synchronization protocol for wireless networks, it depends on support from hardware (i.e. TI CC1000 radio's byte interrupt) for a high-precision message time-stamping. While that kind of support is readily available on commercialized transceivers, it is not trivial on Software Defined Radio (SDR) platforms without modifying their firmware since SDR only acts as a wide-band RF transmitter/receiver. Most of the processing for PHY and MAC layers have to be handled on software layer. On the other hand, TPSN and RBS do not require low level time-stamping to operate. However, TPSN suffers from high overhead as it relies on two-way synchronization exchanges between two nodes. RBS is based on the differences in



(a) Functional block diagram of our implementation in GNURadio. (b) Linux notebook, USRP B200/B200mini, and Galaxy S8+.

Figure 8. MSHCS-MAC implementation and hardware platforms.



(a) Throughput of standard IEEE802.15.4 and MSHCS-MAC. (b) Captured spectrogram.

Figure 9. Single hop performance under PU activity.

receiving broadcast messages from a reference point and works best in a single hop. On the downside, it does not handle delays between the reference point and receivers.

In our proposed MSHCS-MAC, a hybrid time synchronization scheme is designed by adapting RBS as a coarse-grained protocol for devices within a cluster and TPSN as a fine-grained one to synchronize all cluster together (Fig. 7). This scheme maintains low overhead while keeping synchronization error at milliseconds level for our MAC protocol to operate. Since each cluster heads actively broadcast beacons during beacon durations, they are the natural choice for reference points of each cluster in RBS protocol. Reference timestamps are included in each beacon packet. Every device in a cluster uses the received beacon to synchronize with each other. The synchronization error between any two devices (except the cluster head) in a cluster could be kept at microseconds level. While RBS does not handle synchronization errors between cluster head and other devices, proper MAC layer time-stamping can limit error to be within several milliseconds. To handle clock drift, all devices in a cluster will update their time each time they receive a beacon.

TPSN is used to synchronize reference points (i.e. cluster heads) of all clusters together. Its hierarchical structure is also well suited to our MSHCS-MAC's tree topology. SUC can be chosen as the root node of TPSN where other SURs need to synchronize when they join the network. After each SUR has successfully bootstrapped and synchronized with its parent cluster head by beacons, it needs to perform TPSN to achieve better synchronization accuracy before creating its own cluster as shown in Fig. 7b. Clock offset (δ) and transmission delay (Δ) could be calculated after a pair-wise exchange of synchronization request and acknowledgment.

IV. Implementation and Evaluation

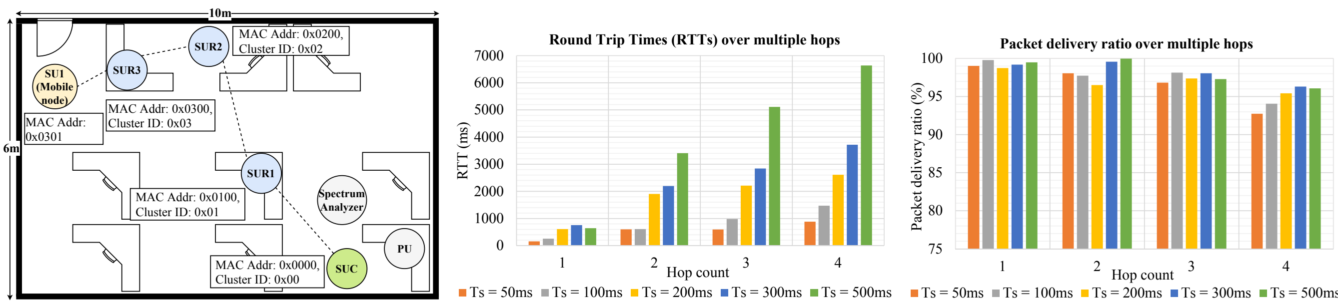
A. Implementation

Our proposed MSHCS-MAC is implemented on GNU Radio as shown in Fig. 8a. We exploit Bloessl's implementation of IEEE 802.15.4 on USRP (Universal Software Radio Peripheral) [17] for our physical layer. The CR-MAC layer holds our MSHCS-MAC and spectrum sensing implementations. On routing layer, simple static routing rules are implemented for evaluation purpose. User's applications can communicate with our MAC protocol through UDP sockets. Linux notebooks and Galaxy S8+s equipped with USRP B200/B200mini devices over USB 3.0 bus are used as evaluation platforms (Fig. 8b).

B. Evaluation

A multihop CRNs testbed has been installed to evaluate for implementation (Fig. 10a). There are 5 CR devices in the testbed: 1 SUC, 3 SURs, and 1 mobile SU node. Notebooks equipped with USRP B200s are used for SUC and SUR nodes, while Galaxy S8+ is used as a mobile SU node. They are placed in a line topology to create a CRN with 4 hops. Additionally, a jammer, which represents PU, and a spectrum analyzer are also set up in our experiment. The area of our testbed is 10x6 m^2. Time synchronization errors among CR devices in our testbed are measured to be within 1-2 ms. Hence, sensing, beacon, reporting durations are set as 10, 5, 5 ms, respectively. Time slot durations are set to be in the range from 50 to 500ms. The number of channels is set to 4 ranging from 2.464 to 2.481GHz. 2MHz bandwidth is allocated for each channel. We measure the maximum throughput between 2 devices under PU activity (i.e. jammer continuously jams the channel which 2 devices are transmitting on), round trip time delay and packet delivery ratio over multiple hops. The results are presented in Fig. 9, and 10.

Fig. 9 presents single-hop throughput of standard IEEE802.15.4 and MSHCS-MAC when a PU continuously jams the transmitting channel. While throughput of standard IEEE802.15.4 dramatically decreases when we increase jamming power, MSHCS-MAC can maintain high throughput by evacuating the jammed channel and continuing its transmission on other channels. Fig. 10 shows measured round trip time delays and packet delivery ratios over 4 hops and with different Ts configurations. Although it has been stated in our previous work [5] that different Ts configurations do not have much effect on single hop performance of SHCS-MAC, they directly affect multihop communication in our proposed MSHCS-MAC protocol. Since SUR performs switching between parent and local clusters' CHS each Ts, smaller Ts will result in smaller round-trip delay over multiple hops as shown in Fig. 10b. However, smaller Ts also poses



(a) Multihop testbed. (b) Round trip time delay. (c) Packet delivery ratio.

Figure 10. MSHCS-MAC performance over multiple hops and with different Ts configurations.

challenges as it requires more precise time synchronization and more computational power. It is shown in Fig. 10c that smaller Ts tends to have lower packet delivery ratios after several hops since devices get overruns with rapid switching and processing large amounts of samples coming from USRP. The mobile node (i.e. Galaxy S8+) experiences more overruns with smaller Ts than CR devices equipped with notebooks.

V. Conclusion

In this paper, we present our proposed mac protocol for multi-hop CRNs based on Slow Hopping and Cooperative Sensing approach, called MSHCS-MAC. The proposed protocol can perform cognitive communication over multiple hops with only one radio is needed. Essential functions such as cooperative spectrum sensing and time synchronization have been also integrated to provide a full-blown CR-MAC protocol. A multihop CRN testbed has been deployed to evaluate its performance and demonstrate the potential for practical uses. As the future works, we are working on improving time synchronization and optimizing our performance on mobile devices.

Acknowledgement

This work has been supported by National GNSS Research Center program of Defense Acquisition Program Administration and Agency for Defense Development.

References

[1] GSMA, "Mobile spectrum - data demand explained," June, 2015.

[2] V. Valenta, R. Maršálek, G. Baudoin, M. Villegas, M. Suarez, and F. Robert, "Survey on spectrum utilization in europe: Measurements, analyses and observations," in *Cognitive Radio Oriented Wireless Networks & Communications (CROWNCOM), 2010 Proceedings of the Fifth International Conference on*. IEEE, 2010, pp. 1–5.

[3] Y. R. Kondareddy and P. Agrawal, "Synchronized mac protocol for multi-hop cognitive radio networks," in *Communications, 2008. ICC'08. IEEE International Conference on*. IEEE, 2008, pp. 3198–3202.

[4] C.-F. Shih, T. Y. Wu, and W. Liao, "Dh-mac: A dynamic channel hopping mac protocol for cognitive radio networks," in *Communications (ICC), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1–5.

[5] Y.-h. Lee and D. Kim, "Slow hopping based cooperative sensing mac protocol for cognitive radio networks," *Computer networks*, vol. 62, pp. 12–28, 2014.

[6] A. De Domenico, E. C. Strinati, and M.-G. Di Benedetto, "A survey on mac strategies for cognitive radio networks," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 1, pp. 21–44, 2012.

[7] H. Su and X. Zhang, "Cross-layer based opportunistic mac protocols for qos provisionings over cognitive radio wireless networks," *IEEE Journal on selected areas in communications*, vol. 26, no. 1, 2008.

[8] B. Hamdaoui and K. G. Shin, "Os-mac: An efficient mac protocol for spectrum-agile wireless networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 8, pp. 915–930, 2008.

[9] C. Cordeiro and K. Challapali, "C-mac: A cognitive mac protocol for multi-channel wireless networks," in *New Frontiers in Dynamic Spectrum Access Networks, 2007. 2nd IEEE International Symposium on*. Ieee, 2007, pp. 147–157.

[10] M. Timmers, S. Pollin, A. Dejonghe, L. Van der Perre, and F. Catthoor, "A distributed multichannel mac protocol for multihop cognitive radio networks," *IEEE Transactions on vehicular technology*, vol. 59, no. 1, pp. 446–459, 2010.

[11] S. Sengupta and K. Subbalakshmi, "Open research issues in multi-hop cognitive radio networks," *IEEE Communications Magazine*, vol. 51, no. 4, pp. 168–176, 2013.

[12] D. Cabric, S. M. Mishra, and R. W. Brodersen, "Implementation issues in spectrum sensing for cognitive radios," in *Signals, systems and computers, 2004. Conference record of the thirty-eighth Asilomar conference on*, vol. 1. Ieee, 2004, pp. 772–776.

[13] A. Sahai, R. Tandra, S. M. Mishra, and N. Hoven, "Fundamental design tradeoffs in cognitive radio systems," in *Proceedings of the first international workshop on Technology and policy for accessing spectrum*. ACM, 2006, p. 2.

[14] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *ACM SIGOPS Operating Systems Review*, vol. 36, no. SI, pp. 147–163, 2002.

[15] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*. ACM, 2003, pp. 138–149.

[16] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The flooding time synchronization protocol," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM, 2004, pp. 39–49.

[17] B. Bloessl, C. Leitner, F. Dressler, and C. Sommer, "A gnu radio-based ieee 802.15. 4 testbed," *12. GI/ITG KuVS Fachgespräch Drahtlose Sensornetze (FGSN 2013)*, pp. 37–40, 2013.