Abstract—Next generation networks are expected to be ultra dense and aim to explore spectrum sharing paradigm that allows users to communicate in licensed, shared as well as unlicensed spectrum. Such ultra-dense networks will incur significant signaling load at base stations leading to a negative effect on spectrum and energy efficiency. To minimize signaling overhead, an ad-hoc approach is being considered for users communicating in unlicensed and shared spectrum. Decision of such users need to be completely decentralized as: 1) No communication between users and signaling from the base station is possible which necessitates independent channel selection at each user. Collision occurs when multiple users transmit simultaneously on the same channel, 2) Channel qualities may be heterogeneous, i.e., they are not same across all users, and moreover are unknown, and 3) The network could be dynamic where users can enter or leave anytime. We develop a multi-armed bandit based distributed algorithm for static networks and extend it for the dynamic networks. The algorithms aim to achieve stable orthogonal allocation (SOC) in finite time and meet the above three constraints with two novel characteristics: 1) Low complex narrowband radio compared to wideband radio in existing works, and 2) Epoch-less approach for dynamic networks. We establish convergence of our algorithms to SOC and validate via extensive simulation experiments.

Index Terms—Multi-player multi-armed bandit, ad-hoc networks, dynamic networks, distributed learning.

I. INTRODUCTION

Next generation wireless networks such as 5G aim to offer the wide range of new services such as enhanced local broadband, high-speed multimedia, mission-critical control, private networks such as Industrial IoT and enterprise [1] via spectrum sharing. Such networks with diverse service requirements are expected to greatly enhance user experience [1]. Recently, 3GPP proposed a new radio (NR) based heterogeneous networks consisting of base stations of various sizes. Compared to existing networks, NRs can operate not only in licensed spectrum but also in the shared (2.3 GHz/3.5 GHz) as well as unlicensed spectrum (2.4 GHz / 5-7 GHz / 57-71 GHz). Such network opens up many interesting challenges such as resource allocation, dynamic and context-aware network adaptation, and in-depth knowledge discovery in the complex environment for which machine learning and artificial intelligence frameworks offer novel solutions [1][4].

The next generation networks are envisioned to work on the principle of separate signaling (large base station) and data infrastructure (small base stations) which allows adaptation of data network to the current traffic situation while maintaining the coverage. These networks will be ultra dense with very high peak rate but relatively lower expected traffic per network node [1]. This makes signaling (control communications) component to be a substantial part of the network traffic leading to a negative effect on the energy and spectrum efficiency. Further, separate signaling and data infrastructures put a significant signaling load on the large base stations, especially in ultra-dense networks. To reduce this, ad-hoc network approach is being considered for users utilizing unlicensed and shared spectrum [1][4]. This not only reduces the signaling load at base stations but also allows the higher number of users per base station (dense networks).

However, the channel selection needs to be done independently at each user since ad-hoc network does not support any direct communication/coordination between users. In addition, channel statistics may not be known which requires it to be learned. In this paper, we explore multi-player multi-armed bandit (MPMAB) framework which enables learning and coordination tasks at each user there by improving the spectrum and energy efficiency of the ad-hoc networks.

MPMAB is a variant of the stochastic multi-armed bandits (MAB) where multiple players/users aim to maximize the sum of their throughputs by transmitting on the same set of channels [5]. Due to hardware and power constraints of battery operated users, we assume that a user can either sense or transmit but not both simultaneously. Furthermore, user can sense or transmit only over single channel in each time slot. We refer such radio terminals as narrowband radios. In ad-hoc networks, the users cannot communicate/coordinate with each other and may not know the number of other users in the network. If two or more users transmit on the same channel simultaneously, they experience ‘collision’ and the packet needs to be transmitted again. Such collisions do not provide any information about the transmission or channel status. In addition to unknown statistics, channels are heterogeneous where the average throughput on a channel may not be the same for all users. Later, we consider the dynamic networks where users may enter or leave without prior agreements. Even though all users employ the same algorithm, new users need to learn to coordinate without any prior knowledge of the network and user status. Such task poses a real challenge in the ad-hoc networks and is the focus this paper.

We develop distributed algorithms for static and dynamic ad-hoc networks that enable users to reach a stable orthogonal configuration (SOC). Under SOC no two users would simultaneously improve their throughput if they swap their channels. Reaching the SOC as early as possible is critical as it allows the users to transmit on one of their preferred channel without incurring significant number of collisions. Our contributions can be summarized as follows:

- For a static ad-hoc network with an unknown number of users and heterogeneous channels, we develop the dSOC_SN algorithm which convergence in finite time to SOC with high probability.
- For a dynamic ad-hoc network with heterogeneous channels and unknown number of users, we develop the dSOC_DN algorithm. We give an high probability upper bound on the time to reach SOC after a user leaves or enters into the network. dSOC_DN algorithm is the first algorithm for the dynamic heterogeneous networks and
is based on novel epoch-less approach (without restarting the algorithm).

- We validate performance of both dSOC_SN and dSOC_DN through extensive simulations. They outperform state-of-the-art algorithms for heterogeneous ad-hoc networks.

- Our algorithms need low complex single antenna narrowband radio compared to wideband radios in existing works such as [18]. Unlike [12,14], the dSOC_DN algorithm does not require global clock synchronization for new users thereby further reducing the algorithm and radio complexity. These advantages make our algorithms suitable for completely decentralized and battery operated users.

The paper is organized as follows. The related work and network model are discussed in Section II and Section III, respectively. The proposed algorithms and their analysis are presented in Section IV and Section V for static and dynamic networks, respectively. The simulation results are presented in Section VI and Section VII concludes the paper.

II. RELATED WORK

Various works dealing with coordination in multi-user ad-hoc networks have been discussed in literature. In this section, we focus on works employing the multi-armed bandit (MAB) based approach for channel selection as it outperforms other approaches employing random hopping based techniques. We use the notations $N$ and $K$ to denote the number of users and channels, respectively.

The $\rho^{rand}$ [5] is one of the first distributed algorithm for static homogeneous ad-hoc network with known number of user. It uses well known upper confidence bound (UCB) based algorithm to learn the channel statistics and channel selection. The UCB is combined with rank based randomization approach to orthogonalize users in the best $N$ channels. Though $\rho^{rand}$ offers asymptotic logarithmic regret (i.e. throughput loss), it incurs large number of collisions due to random reordering where users randomly select new channel after every collision which in turn may results in collision with other users. Subsequent algorithms in [6,9] are based on $\rho^{rand}$ and they offer further improvement in performance by reducing the number of collisions among users. The musical chair based MCTopM algorithm in [8] is the current state-of-the-art algorithm for static homogeneous ad-hoc network with known $N$. In MCTopM, when two users collide, users switch to new channel only when current channel is non-optimal and user is not locked. This allows faster orthogonalization after collision. Another algorithm in [10] assumes undetectable collisions, i.e. user does not know whether transmission failure is due to poor channel or due to collision with other users. Recent algorithm in [10] considers static heterogeneous ad-hoc network and it is the first algorithm that guarantees poly-logarithmic regret. However, major drawback of these algorithms [5,10] is that they need prior knowledge of $N$, which is unrealistic in distributed ad-hoc networks.

Recently, various algorithms [11-18] have been proposed which do not need the prior knowledge of $N$. Among them, algorithms in [11,14] consider only homogeneous channels while algorithms in [16,18] consider homogeneous as well as heterogeneous channels. The works in [11] and [12] consider dynamic networks with homogeneous channels and unknown $N$. The MEGA algorithm in [11] uses the classical $\epsilon$-greedy MAB algorithm and ALOHA based collision avoidance mechanism. Though collision frequency reduces in MEGA as the game proceeds, it may not go to zero as shown in [12]. To overcome this [12] develops MC algorithm that incurs collisions due to random hopping (RH) in the initial learning phase and guarantees collision-free access over optimum channels subsequently. Though MC performs better than MEGA, its performance in the learning phase is poor – MC uses collision information to estimate $N$ and forces a large number of collisions to get a good estimate. In [14], Secondary user Co-ordination with Fairness (SCF) algorithm for static network with unknown $N$ is proposed and it is extended it to dynamic networks (dynamic SCF i.e., DSCF) using epoch approach, where the algorithms restart at certain intervals. For dynamic networks, though DSCF algorithm outperforms MEGA and dynamic MC (DMC) algorithms, use of epoch approach makes it far from being optimal.

To overcome the above drawbacks, trekking based algorithm for static network (TSN) in [13] and for dynamic network (TDN) in [15] are recently proposed. As opposed to existing algorithms which separates estimation and orthogonalization tasks, it is shown that knowing $N$ is not necessary to find an optimal channel allocation. However, these algorithms fail when channel statistics are heterogeneous. Furthermore, algorithms in [12,15] need prior knowledge of minimum gap in channel statistics which is generally unpredictable.

Among the algorithms which consider heterogeneous channels, DE3 algorithm in [17] employs Bertsekas auction algorithm which requires the users to convey the bids frequently to claim their preferred channels. Such assumption is not desirable in next-generation communication networks due to signaling overhead, intentional jamming attacks and vulnerability to noise. The Coordinated Stable Marriage MAB (CSM_MAB) algorithm in [18] overcomes the need for direct communications but requires more observations in each time slot. Specifically, it requires that all users to simultaneously sense all channels (wideband sensing). As discussed later, such radio terminals are computationally complex and may not be suitable for low cost, battery operated resource constrained user terminals. Recently, CSM_MAB is extended to include the dynamic networks using wideband sensing, however it puts restriction on when a new user can enter.

Our work deals with heterogeneous channels and considers both static and dynamic networks. Each user need to sense and transmit only one channel at a time (narrowband sensing) which is more realistic and computationally efficient. Our novel frame structure makes the protocol simple and easy to implement and achieves performance better than that achieved by CSM_MAB with wideband sensing.

A. Radio Models

One of the major aspect of distributed algorithm is the capability of radio terminal. Existing distributed algorithms consider various types of radio architectures which not only impact learning period but also the complexity and performance of the distributed algorithm. These architectures offer a trade-off between sensing/transmission capability and implementation complexity. For instance, [19] considers sophisticated radio terminals with two independent analog signal processing (ASP) blocks each consisting of an antenna, matching units, amplifiers, analog-to-digital or digital-to-analog converters, etc. One ASP block is used for narrowband (single channel) transmission while second ASP block can sense all channels simultaneously, i.e. wideband sensing. Such wideband sensing makes an estimation of $N$ trivial and simplifies coordination
since users can differentiate between users on different channels. However, the wideband channel sensing needs high-speed ADCs making ASP as well as subsequent digital baseband processing complex, power hungry and hence, not suitable for battery operated radio terminals [19]. The non-contiguous wideband channel sensing is even more challenging. Another architecture consisting of two narrow-band ASP blocks which allow simultaneous transmission and sensing over different channels have been considered in [17]. In this paper, we consider the architecture which has the lowest complexity among the three. It consists of single narrowband ASP chain which allows either transmission or sensing over a single channel in a given time slot. Such architecture can detect the presence of another user on their channel either by experiencing collision or sensing but cannot estimate the number of collided or sensed users. Furthermore, architecture cannot sense multiple channels simultaneously making the estimation of $N$ and establishing coordination extremely challenging than in [17] [18].

III. NETWORK MODEL

Consider an ad-hoc network consisting of $N$ users competing for $K$ ($\geq N$) channels in an unlicensed spectrum. We assume the communication is time slotted, and the users are clock synchronized with respect to beginning of each time slot as in [5][7][8][12]. In each time slot, each user can transmit only once over any one of the $K$ channels. When two or more users transmit simultaneously on a channel, a collision occurs and all the uses involved in the collision need to re-transmit the lost packet. The users are not aware of how many other users are present ($N$ is unknown) and no central coordinator exists to facilitate their channel selections.

Another major characteristic of our network is that the channels are heterogeneous, i.e., the expected reward/throughput on a channel depends not only on the channel but also on the user selecting it. Such model is more practical than homogeneous channels as it considers location dependent channel conditions. Let $\mu_{n,k} \in [0,1]$ denote the expected reward for user $n \in [N]$ on channel $k \in [K]$ on a collision free transmission. These mean values are unknown to the users and user $n$ can only observe $\{\mu_{n,k}, k \in [N]\}$, i.e., all observations are local and a user does not know the expected reward offered by the channels to other users. The reward observed by an user on a channel under collision free transmissions are assumed to be independently and identically distributed. The same setup is also considered in [17][18].

The performance of the distributed algorithms is compared in terms of expected rewards/throughput. The maximum reward is achieved when all users are on orthogonal channels and channel allocation guarantees maximization of the sum of rewards over all users. Formally, let $\pi : [N] \rightarrow [K]$ denotes an orthogonal allocation of the users to the channels and $C$ denotes all such possible allocations. Then, the maximum expected reward is given by

$$R_{\text{max}} = \max_{\pi \in C} \sum_{n=1}^{N} \mu_{n,\pi(n)}.$$  \hfill (1)

Achieving $R_{\text{max}}$ requires all users to know the expected reward of all channels at all users. This requires all the users share their observation with all other users in the networks which either needs direct communication between users or sophisticated signaling scheme. Instead, we focus on achieving a stable orthogonal allocation configuration (SOC) [18] where no two user can simultaneously agree to swap their channels without one of them getting a ‘less preferred’ channel than its current one. To explain SOC, we first define the rank of a user which corresponds to the number of channels whose expected reward is higher than the current channel the user has selected. For the $n$-th player, it is given by,

$$\gamma_n(t) = \sum_{k=1}^{K} 1\{\mu_{n,k} > \mu_{n,\pi_n(t)}\},$$  \hfill (2)

where $\pi_n(t)$ indicates the channel selected by user $n$ in time slot $t$ using policy $\pi : \{\pi_n : t \geq 1\}$. The total rank of the network (also called network potential) is given by

$$\gamma_n(t) = \sum_{n=1}^{N} \gamma_n(t).$$  \hfill (3)

An assignment $\pi$ is said to be SOC if a swap of channels between any pair of users or switch to the vacant channel do not strictly decrease network potential. For a network in SOC, user will have no incentive to request any other user for a swap of their channels, hence channel switches/ swaps will not occur. Our aim is to design distributed algorithms that converge to a SOC as early as possible. We note that there could exist multiple SOCs.

IV. STATIC NETWORK: ALGORITHM AND ANALYSIS

In this section, we consider the static network where all users simultaneously enter into the network at the beginning ($t = 0$) and remain active until the end. We describe an algorithm named distributed Stable Orthogonal Configuration for Static Network ($dSOC\_SN$) and analyze its performance.

A. $dSOC\_SN$ Algorithm

The algorithm is run independently at each user and it comprises of two phases: 1) Random hopping (RH), and 2) Sequential master and channel switching (SMCS). The pseudo code is given in Algorithm 1 where $K$ indicates the number of channels and $\pi_{T_{rh}}(n)$ indicates the channel on which user $n$ gets locked in the RH phase.

Algorithm 1 $dSOC\_SN$ Algorithm

Input: $K$
$\pi_{T_{rh}}(n) = RH(K)$
SMCS ($\pi_{T_{rh}}(n), K, 1$)

1) Random Hopping (RH) Phase: The RH phase allows users to orthogonalize on different channels. This phase is an adaption of the RH phase in [13] where licensed spectrum is considered as opposed to unlicensed spectrum here. The pseudo code of the RH phase is given in Subroutine 1. In RH phase, each user selects a channel drawn uniformly at random (line 7) in each time slot. Once the user observes a collision-free transmission on a channel, user locks on that channel and plays it till the end of the RH phase (lines 9–11). Such channel is referred to as reserved channel, $\pi_{T_{rh}}(n)$. The duration of the RH phase is fixed and is set equal to $T_{rh}$ (See Lemma 1).
Subroutine 1: Random Hopping (RH)

1: Input: $K$
2: Set $Lock = 0$ and compute $T_{rh}$ using Eq. [5]
3: for $t = 1 \ldots T_{rh}$ do
4: if ($Lock = 1$) then
5: Choose channel, $\pi_n(t) = \pi_n(t - 1)$
6: else
7: Randomly choose channel, $\pi_n(t) \sim U(1, \ldots, K)$
8: Set $Lock = 1$ if no collision is observed.
9: end if
10: end for
11: Return $\pi_n(T_{rh})$ indicating the channel index at $t = T_{rh}$.

2) Sequential Master and Channel Switching (SMCS) Phase: All users enter into the SMCS phase at time $t = T_{rh} + 1$ and continue in that phase till the end. In this phase, the users maintain a list of their preferred channels, i.e., the channels that are better than their current one, and taking turns, request other users who are currently on one of their preferred channels to switch their channel with them. If a preferred channel happens to be un-occupied, it is simply taken, otherwise the user gets it only if the other user accepts the switch request. The other user accepts the request only if the channel she will be shifting to is also one of her preferred channels, otherwise she rejects the request. If the switch request is rejected, the user try her next preferred channel. To allow such negotiations, we divide the time slots into blocks which further consists of sub-blocks. Learning and channel switching happen in the SMCS phase. We first discuss the signaling scheme required for user to switch their channel and then the learning method.

B. Signaling for Channel Switching

The SMCS phase consists of a sequence of one hot switching (OHS) blocks each of $T_{ohs}$ time slots that repeat one after another. Each OHS block consists of $K$ master blocks (MB) and each master block is made up of $T_{mb}$ time slots. Fig. 1 gives structure of the OHS and master blocks. The duration of each OHS block is $T_{ohs} = KT_{mb}$ slots.

![Diagram of OHS and Master Blocks]

Fig. 1: Different phases and sub-blocks of the dSOC_SN algorithm for static ad-hoc network.

Each user uses the channels they get at the end of the RH phase as their reserved channel and attempts to get a channel that is better than her current reserved channel by requesting switch. This is achieved by allowing each user to become a master in a specific MB. At any time, only one user is allowed to be a master while all other users continue to transmit on their respective reserved channels and are referred to as non-masters. When a user becomes a master, she requests other users for a switch of channels that are better than her current channel. If she gets a better channel, she continues to transmit on it till she becomes the master again. If she moves to a new channel, it becomes her reserved channel, otherwise the current channel will be her reserved channel.

In the $k$th MB, the user on the $k$th channel becomes the master and gets a chance to move to a better channel by sending switching requests on channels that are better than her current channels. This is facilitated by a signaling scheme defined as follows: The MB block consists of $K$ sub-blocks (SB) of two slots each (See Fig. 1(c)). The first slot in each SB is reserved to as channel transmit (CT) and the second as channel switch (CS). In the $i$th MB, each user selects her $(i - 1)$th preferred channel and transmits. If a collision is incurred in the CT slot, then it implies that no user is present on that channel and the master switches to it and makes it her reserved channel (scenario 3). If a collision is observed in a CT slot, it implies that another user is present on that channel and the master transmits on the same channel in the next CS slot. If the master does not encounters a collision in the CS slot after the collision in previous CT slot, it implies the current user on that channel is not willing to exchange her channel and rejected master’s request to switch channels (scenario 3). If the master encounters a collision in the CS slot, it implies that the user on the current channel accepts master’s request for channel switch (scenario 1), in which case the master switches to the requested channel and makes it her new reserved channel. These and other possible scenarios are summarized in Fig. 2. If the master gets a new channel no more switch are requested in the current MB, otherwise the process is repeated in the next SB for the next preferred channel.

![Table of Signaling Scenarios]

Fig. 2: Various channel switching scenarios between master and non-masters in a given CT/CS time slots. Note that Tx indicates transmission and X indicates don’t care.

C. Structure of Master Block

Note that the block structure allows the users to identify the reserved channel of the master in each MB which helps them to decide whether to accept or reject a request to switch. Specifically, if an user receives a switch request in the $k$th MB, then she knows that she will move to $k$th channel on accepting the request.

Since there are $K$ channels, a master can switch to any one of the $(K - 1)$ channels which means the duration of MB can be set at most $2(K - 1)$ slots. However, in $dSOC_SN$ we force the master to transmit on its reserved channel in the first sub-block and thereafter follows its preference list in the rest of the $(K - 1)$ sub-blocks. Hence, the duration of each MB is $T_{mb} = 2K$ slots. As we will see later, this block structure

10-th preferred channel corresponds to her current reserved channel.
allow new users to synchronize with the existing users in the dynamic networks considered in Section IV.

D. Learning the Channel Statistics

As discussed in Section III users are not aware of the expected rewards they receive on each channel and need to learn and index (or rank) them. We employ the multi-armed bandit (MAB) approach based learning algorithm to find index for each channel. Various indexing methods based on Upper Confidence Bound (UCB), Bayes-UCB algorithm, Thompson Sampling (TS) can be used [20]. In this paper, we focus on indexing based on UCB and its analysis. However, our algorithm can work with other indexing method as well. The UCB algorithm is based on exploration-exploitation trade-off and its indexing is based on the optimistic estimates of mean rewards. For the n user, the channels are indexed based on the UCB scores given as follows [20].

\[ Q_{n,k}(t) = \frac{P_{n,k}(t)}{S_{n,k}(t)} + \sqrt{\frac{2 \log t}{S_{n,k}(t)}} \quad \forall k \in [K], \quad (4) \]

where \( P_{n,k} \) is the total reward received by user \( n \) on channel \( k \) when it was chosen for \( S_{n,k} \) time slots. The UCB algorithm is asymptotically optimal in the sense that it gives higher score sub-optimal channel exponentially smaller number of times compared to the optimal channel.

The pseudo-code of the SMCS phase is summarized in Subroutines 2, 4, 5. \( \pi_{n,r} \) indicates the reserved channel of the user \( n \) and it is the channel on which the user had recently locked. For example, \( \pi_{n,r} = \pi_{r,n}(n) \) when user enters in the SMCS phase. After entering the SMCS phase, user can either be in master or non-master mode based on the index of the reserved channel (lines 4 – 8: Subroutine 2). In the master mode (Subroutine 3), master identifies the index of the SB. As mentioned before, in the first SB of each MB, the master transmits on its reserved channel (line 2: Subroutine 2). In the rest of the SB blocks, master selects the channel based on the preference list obtained using USB algorithm (line 4: Subroutine 3). As discussed before, master moves to non-master mode in one of the three scenarios: 1) No collision in CT slot (lines 5 – 6: Subroutine 3), 2) Collisions in CT and subsequent CS slots (lines 7 – 8: Subroutine 3), and 3) End of the MB block (lines 2: Subroutine 3).

The pseudo-code for channel selection in non-master mode is given in Subroutine 4. If the non-master faces collision in the CT slot, it checks whether the reserved channel of the master (i.e., the channel with index MB) is better than her current channel. If yes, non-master accepts the switch request by transmitting in the subsequent CS slot, updates and moves to the new reserved channel (lines 8 – 10: Subroutine 4). Otherwise, she remains silent in the CS slot indicating switch reject (line 12: Subroutine 4). If there is no collision in the CT slot, non-master transmits on the reserved channel in the CS slot (line 6: Subroutine 4).

E. Analysis

In this section, we analyze the performance of \( dSOC_{SN} \) algorithm and show that it leads to a stable orthogonal configuration (SOC). Our main result is the following theorem.

Theorem 1: Consider a network with \( K \) channels and \( N \) users with channel rewards characterized by \( \{\mu_{n,k}\} \) for all \( n \in [N] \) and \( k \in [K] \). For any \( \delta > 0 \), set \( T_r(\delta) \) as in Eq. [5] Then, there exists \( T(\delta) \) such that for all \( t \geq T_r(\delta) + T(\delta) \), the probability of the network being in an SOC is at least \( 1 - 2\delta \).

We prove the result using the following lemma. Its proof is given in the appendix.

Lemma 1: Let \( \delta \in (0, 1) \). If RH sub-phase is run for \( T_r(\delta) \) number of time slots, then all the users will orthogonalize with probability at least \( 1 - \delta \) where

\[ T_r(\delta) := \left\lceil \frac{\log(\delta/K)}{\log(1 - 1/4K)} \right\rceil \quad (5) \]

The lemma guarantees that the users are orthogonalized with probability at least \( 1 - \delta \) at the end of the RH sub-phase. We next prove the theorem conditioned upon this event. The proof is an adaptation of the proof of Thm 1 in [13] to our specific block structure.

Outline of Proof of Thm. [2] Note that the block structure is designed such that even if user’s request to swap on a channel is rejected, she still gets to observe a reward/throughput sample from that channel (in CS slot). Thus the master gets to explore

---

Subroutine 2: SMCS Phase

1: Input: \( \pi_{n,r}, K, M_{ind} \)
2: for \( OHS=1,2,..... \) do
3: \#master block counter
4: \#first SB
5: for \( MB = M_{ind}, 2 \ldots K \) do
6: \# master mode.
7: Enter into Master mode.
8: else
9: Enter into non-Master mode.
10: end for

Subroutine 3: Master Mode

1: Input: \( \pi_{n,r}, K \)
2: Transmit on \( \pi_{n,r} \) for two time slots.
3: for \( SB=1,2,....K-1 \) do
4: Transmit over \( SB^{th} \) channel in the preference list for two slots (CT and CS).
5: if No Collision on CT slot then \#Vacant channel
6: Switch to current channel and enters into the non-Master mode.
7: else if Collisions on CS and CT slots then
8: Switch to current channel and enters into the non-Master mode.
9: end if
10: end for

Subroutine 4: Non-Master Mode

1: Input: \( \pi_{n,r}, K, MB \) (Current channel of master)
2: for \( SB=1,2,.....K-1 \) do
3: if Collision observed in CT slot then
4: Decide if like to switch with the channel MB.
5: else
6: Continue on the current channel.
7: end if
8: if Switch request accepted then
9: Transmit in the CS slot and shift to channel MB.
10: Update \( \pi_{n,r} = MB \).
11: else
12: Do not transmit in the CS slot.
13: end if
14: end for
the channels as if she is the only user in the network. Also, each user gets to observe many samples of each channel as their ranking of the channel is based on the UCB Index given in Eq. 4. Then, following the same arguments as in the proof of Lemma 1 in [13], if all channels are sampled at least $s_{\min}$ number of times, then all the users will have correct ranking of the channels (with respect to the true means) with probability at least $1 - 2r^{-4}$. Thus, any channel switch thereafter results in both users moving to their better channels leading to decrease in network potential. The $s_{\min}$ is given by

$$s_{\min} := \frac{8 \log t}{\Delta_m^2}$$

(6)

where

$$\Delta_m = \min_{n \in [N]} \Delta_n$$

and

$$\Delta_n = \min_{i,j \in [N], i \neq j} |\mu_{n,i} - \mu_{n,j}|$$

However, for the specific block structures designed for coordination, each user may not get an informative reward sample in each time slot. A master in a MB, will get at least 2+K-1 samples, while a non-master will get at least 2K-2-2 samples. Hence over an OHS block consisting of 2$K^2$ slots, each user will get at least $(K-1)(2K-4) + K + 1$ informative samples. Also, the minimum number of samples given in Eq. 6 should be satisfied for all channels. Hence, the condition on smallest $t$ such that Eq. 5 holds across all channels is given by

$$t \geq \frac{2K^2}{(K-1)(2K-4) + K + 1} s_{\min} \geq \frac{16K}{\Delta_m^2} \log t.$$  

Again using [13][Lemma 1], we can show that the smallest $t$, denoted $t_m$, satisfying Eq. 6 is finite and satisfies

$$t_m \leq \frac{M - 1 - \sqrt{(M-1)^2 - 4M}}{2},$$

where $M := \frac{16K}{\Delta_m^2}$. Thus, for all $t \geq t_m$ every channel switch results in both users switching to better channels and the network potential decreases.

Further, note that the maximum value of the network potential is at most $N(K-1)$ and user becomes master and gets a chance to switch within every 2$K^2$ slots. Since every user becomes a master in each OHS, it takes at most $K-1$ OHS slots to reach stable allocation after $t > t_m$. Then, probability that the network is in SOC within $\tau = 2KN(K-1)$ slots after initial $t_m$ slots is

$$P_s = (1 - 2r_m^{-4})^{N(K-1)}.$$  

For any $t$, let $S_t = 1$ and $S_t = 0$ denote the events that network is in SOC and not in SOC, respectively. We have

$$\Pr\{S_{t+m} = 1|S_m = 1 \} \geq P_s.$$  

Then, for any $\tau + t_m$

$$\Pr\{S_{\tau} = 0|S_m = 0 \} < (1 - P_s)\frac{T-t_m}{\tau}.$$  

Setting $(1 - P_s)\frac{T-t_m}{\tau} \leq \delta$ and solving we get

$$T(\delta) := t_m + \tau \log \left( \frac{\delta}{1 - P_s} \right)$$

(7)

Then, for all $T \geq T(\delta)$ the network will be in a SOC with probability at least $(1 - \delta)$. Taking into account the initial $T_{\delta}(\delta)$ rounds of random hopping in which orthogonalization happens with probability at least $(1 - \delta)$, we conclude that for all $t > T_{\delta}(\delta) + T(\delta)$, the network will be in SOC with probability at least $1 - 2\delta$. This completes the proof.

V. DYNAMIC NETWORK: ALGORITHM AND ANALYSIS

In this section, we will study dynamic networks where users can enter or leave the network anytime without prior agreement. Existing algorithms for dynamic ad-hoc networks assume global synchronization which means new users have complete knowledge about the status of the network [11] [12] [14], for example, the current MB and its beginning slot. The algorithms in [12] [14] exploit the full knowledge of network state and restarts at regular intervals to account for the dynamic users [11]. However, requiring complete knowledge of the network state is restrictive as non-active users need to continuously sense the network without utilizing the energy efficient sleep mode. We get rid of such assumption by allowing users to identify the parameters of the block on their own. Our frame structure is designed in such a way that the new users can figure out the current state of the network themselves within few rounds.

A. dSOC_DN Algorithm

When a new user enters into the network, she does not have any knowledge about the current MB and the slot type (CT/CS). Her first task then is to synchronize with the network to identify parameters such as index of the MB and know which slot is CT and CS. In an ad-hoc network where there is neither a central controller nor control channel between users, synchronization is a difficult task unless existing users help or guide the new users. To achieve synchronization among the users without the need of global clock and horizon synchronization, we develop the dSOC_DN algorithm with appropriate modifications to the dSOC_SN algorithm. The pseudo-code of the dSOC_DN algorithm is given in Algorithm 2. It consists of two phases: 1) Synchronization phase, and 2) SMCS phase. The synchronization phase enables users to identify block parameters such as $M_{ind}$ and its reserved channel, $\pi_{n,r}$ while SMCS phase is same as dSOC_SN algorithm with two modifications discussed below that aids the synchronization.

Algorithm 2 dSOC_DN Algorithm

Input: $K$

$M_{ind}, \pi_{n,r} = SP(K)$

SMCS ($\pi_{n,r}, K, 1$)

B. Modified SMCS Phase

Recall that in the dSOC_SN algorithm all active users transmit on their reserved channels in the first sub-block of each MB. In the modified SMCS phase, only master transmits on the reserved channel while other users remain silent in the first sub-block of each MB. The first sub-block of each MB is referred to as synchronization sub-block (SSB) as it will help new users to synchronize in the network. For illustration, if a new user observes that no transmission happen for two consecutive time slots on an occupied channel, then she know that the first slot where no transmission happened is the CT slot. Then, she can identify the channel on which transmission occurs and stop listening to that channel. The second modification prohibits the users to leave the network when they are in the non-master mode. Specifically, when an user has to leave she will do so only at the start of the MB where she is supposed to be the master. Without such restriction, the new user will not have sufficient information to identify the block parameters using SSB and she will have
to frequently switch to other channels whenever existing user leaves the network at arbitrary times. Note that the learning users has to delay its departure by at most one OHS duration, i.e., is $2K^2$ slots, which is a vanishing portion of the horizon size and hence this assumption is not overly restrictive. This restriction is applies only to the leaving users and the new users can enter any time.

### C. Synchronization Phase

New user starts with the synchronization phase after entering into the network. In this phase, new user randomly selects a channel in each slot till it finds a channel that is occupied by another user. Once the new user finds an occupied channel, it stays on that to find the network status. We refer the channel that the new user uses to find network state as 'piggyback channel' and the user on the piggyback channel as 'piggybacking user'. We say that new user has entered into the piggyback phase once she find an occupied channel. In piggyback phase, user senses the same channel continuously till it observes no transmissions for two consecutive time slots immediately followed by at least one transmission. After identifying such time slots (i.e., SSB), user can easily differentiate between CT/CS time slots. However, new user cannot know the index of the MB block, $M_{ind}$, without which it cannot enter the SMCS phase.

In order to find $M_{ind}$, new user has to sense the piggyback channel until one of the two events happen: 1) Piggyback user becomes master, or 2) Piggyback user leaves the network. When piggyback user becomes master, new user can sense transmissions instead of silent SSB. Similarly, when piggyback user leaves the network, new user will sense silent slots for at least $2K$ time slots. In each case, $M_{ind}$ is same as the index of piggyback channel. Note that both these events can happen only once in the OHS block and since the duration of OHS block is $2K^2$ time slots, new user must sense the piggyback channel for $2K^2$ time slots in the piggyback phase. After that, new use is guaranteed to have estimated $M_{ind}$ and can enter into the SMCS phase.

After synchronization and before entering into the SMCS phase, the new user needs to have its own reserved channel. It is identified by sequentially sensing the channels until she finds a vacant channel which is not occupied by any of the active users. Note that new user cannot take the reserved channel of the current master and this can be easily avoided as she has complete knowledge of block parameters.

Next, we demonstrate the switching from synchronization to SMCS phase using suitable example. For illustration, we highlight the channel selection of various users during certain interval of the horizon, say $t = 72490$ to $t = 72750$. As shown in Fig. 3, x-axis represents time, y-axis represents the channel index and a number inside the circle indicates particular action. The index of the MB is shown using red colored dark circles and OHS regions are indicated with the different colors. For instance, action 3 shows the boundary between OHS blocks. The Fig. 3 begins with fifth MB of OHS block and comprised of one complete OHS block followed by four MBs of the next OHS block. There are three users (U1, U2, and U3) in the beginning and their channel selections are indicated using different lines. Action 1 indicates entry of new user, U4, at $t = 72500$. After entering into the network, U4 selects channel 5, 7 and 3 uniformly random and senses it. Once she senses the channel 3 as occupied, she enters into the piggyback phase. The duration of the piggyback phase is indicated using yellow shaded region (action 2). As soon as U3 becomes a master, U4 completes it piggyback phase (action 4), identifies the channel 2 as its reserved channel and enters into the SMCS phase. You can also observe the channel swapping or switching between master and non-masters at different instants in Fig. 3. For example, action 5 indicates exit of U1 when she is master while action 6 indicates the channel switching between U3 and U4. Similarly, in the last MB, U2 switches to the channel vacated by U1. In this way, proposed algorithms allows the network to reach SOC within finite time after every entry or exit of an user.

The pseudo code of the proposed synchronization phase is given in Subroutine 5. When a new user enters into the network, she selects the channel uniformly randomly (line 5) and senses it. The user enters into the piggyback phase if the channel is sensed as occupied (line 6). When user senses the channel as vacant for two consecutive time slots, user is said to be synchronized ($Sync = 1$) (line 9). Thereafter, user senses the channel for at most $2K^2$ time slots to identify the $M_{ind}$ (line 12) and enters into the SMCS phase after identifying the reserved channel (line 13).

### Subroutine 5: Synchronization Phase for New User

1: Input: $K$

2: Set Piggyback = 0 and Sync = 0.

3: while Sync == 0 do

4: while Piggyback == 0 do

5: Sense randomly chosen channel, $\pi_n(t)$~$U(1,..,K)$.

6: Enter into Piggyback phase, Piggyback = 1, if channel is occupied

7: end while

8: Sense the same channel, $\pi_n(t)$ = $\pi_n(t-1)$.

9: Synchronization done, Sync = 1, when channel is sensed as vacant for two consecutive time slots followed by at least one transmission.

10: end while

11: Sense the same channel for $2K^2$ time slots.

12: Identify the index of the MB.

13: Identify reserved channel and enter into Subroutine 2: SMCS phase.

### D. Analysis

Next, we analyze the performance of $dSOC_DN$ algorithm and show that it leads to a stable orthogonal configuration in finite time after entry or exit of users. Our main result is the following theorem.
Theorem 2: Consider a network with \( K \) channels and \( N \) users with channel re-wards characterized by \( \{m_{n,k}\} \) for all \( n \in [N] \) and \( k \in [K] \). For the network in SOC, if \( e \) and \( l \) are the number of users enter or leave the network, respectively, then after \( T^d + T^d(\delta) + T^d \) time slots from the recent entry or exit event, the network will be back in SOC with probability \( \delta \) where \( \delta = (0, 1) \).

We prove the result using the following lemmas. Their proofs are given in the appendix.

Lemma 2: In a network consisting of \( K \) channels and \( N \) users, the new user will need at most \( T^d = K(2K + 4) + 1 \) number of time slots to complete the synchronization phase, identify the reserved channel and begin SMCS phase.

Lemma 3: For a network in SOC and \( \delta \in (0, 1) \), when new user begins its SMCS phase, the network will be in SOC again with probability \( \delta \) after \( T^d(\delta) \) number of time slots assuming no user enters or leaves the network where

\[
T^d(\delta) = t_{nu}^d + \log \left( \frac{\delta}{1 - p_{nu}^d} \right)
\]

where

\[
t_{nu}^d = \frac{M_{nu}^d - 1 - \sqrt{(M_{nu}^d - 1)^2 - 4M_{nu}^d}}{2}
\]

\[
M_{nu}^d := \frac{16(K - N)}{\Delta_{\min}^2}
\]

and

\[
p_{nu}^d = 1 - 2(t_{nu}^d)^{-4}
\]

Lemma 4: For a network in SOC and \( \delta \in (0, 1) \), when one of the user leaves the network, the network will be in SOC again in at most \( T^d = 2K^2(K - 1) \) number of time slots provided that no new user enters or leave the network.

Outline of Proof of Thm. 2: The time required for the network to be in SOC depends on the duration of three events: 1) Time required for a new user to enter into the SMCS phase, \( T_s \) (Lemma 2), 2) Time required for new user to learn channel statistics and minimize the network potential, \( T^d(\delta) \) (based on Lemma 3), 3) Time required for the network to reach SOC after an exit of the user, i.e. 2\( K^2(K - 1) \) time slots (Lemma 4).

To find \( T^d(\delta) \), we do following modifications in Lemma 3 and Theorem 1. We replace \( M_{nu}^d \) with \( M^d := \frac{16(K - N)}{\Delta_{\min}^2} \) where \( l \) is the number of users left the network. Similarly, we replace \( p_{nu}^d \) with \( p^d \) such that \( p_{nu}^d = (1 - 2e^{-d})^{e(K - 1)} \) as there are \( e \) new users and maximum possible decrease in potential can be \( e(K - 1) \). Here, \( p^d \) indicates the probability that the network is in SOC within \( T^d = 2Ke(K - 1) \) time slots after initial \( t^d \) time slots from the slot \( e \)-th user enters into the network. Based on these modifications, we get

\[
T^d(\delta) = t^d + \tau^d \log \left( \frac{\delta}{1 - p^d} \right)
\]

Then, in \( T^d + T^d(\delta) + T^d \) time slots after recent entry or exit, the network will be in a SOC with probability at least \( (1 - \delta) \). This completes the proof.

E. More Users than Channels

When number of users are more than the number of channels, i.e., \( N > K \), there can be two possible options in ad-hoc networks: 1) Allow all users to enter into the network using virtual channels, and 2) Restrict some users from entering the network (or in dynamic networks, users can attempt to enter into the network after certain intervals). Existing algorithms such as \([5, 6, 8, 12, 13]\) fail when \( N > K \). Virtual channels are used in \([16]\) which requires a central controller to include \((N - K)\) virtual channels for collision-free sequential hopping. Furthermore, as shown in \([13, 14]\), it is not an efficient algorithm when \( N < K \).

The dSOC_SN algorithm handles \( N > K \) scenario by offering the second option in the RH phase. When an user gets locked on the channel in the RH phase, it enters into the SMCS phase after \( T_{rh} \) time slots, i.e. at the end of RH phase, and those who do not lock on a channel can leave the network in at most \( T_{rh} \) time slots. By the end of the RH phase, \((N - K)\) users will experience continuous collisions within the RH phase they get to know that all the channels and utilized hence they can leave.

In dynamic networks, users can re-enter the network after certain interval which depends on the rate at which users enter or leave the network. In each case, new user needs to complete synchronization phase and identify the reserved channel before entering into the SMCS phase. New user cannot find the reserved channel when \( N > K \). Thus, in \( T_f \) (See Lemma 2) time slots after identifying the \( M_{ind} \), new user can realize the unavailability of the reserved channel and leaves the network. As long as users remain active in the network for short duration, new users can enter the network whenever channels are available. In this way, our algorithms handle the case of \( N > K \) without compromising on the stability of the network.

VI. Experimental Results

To demonstrate the effectiveness of proposed algorithms, we present the simulation results for comparison with respect to parameters such as: 1) Network potential, 2) Average and total reward/throughput, 3) Number of channel switching, and 4) Number of collisions. Initially, we consider \( K = 10 \) and \( N = \{5, 10\} \). Each numerical result presented here is obtained after averaging over 100 independent experiments and the horizon size is \( 100000 \) time slots. The channel statistics are unknown, heterogeneous and chosen randomly in each experiment.

A. Static Network

For static networks, the performance of the dSOC_SN algorithm is compared with the state-of-the-art Coordinated Stable Marriage Multi-Armed Bandit (CSM-MAB) algorithm in \([18]\) and optimal algorithm where channel statistics are known in advance and users are orthogonalized in collision-free manner to achieve optimum reward/throughput. Note that CSM_MAB needs wideband sensing receiver consisting of two parallel ASP blocks and computationally intensive digital baseband processing algorithms compared to single antenna and ASP based narrowband radio for the proposed algorithms. We also consider heuristic dSOC_SN, referred to as dSOC_SN_H, which differs from dSOC_SN in two ways: 1) The size of the MB is reduced to \( K \) time slots instead of \( 2K \) time slots, and 2) User avoids the particular channel for certain interval whenever switch request for that channel gets rejected. The interval doubles after every rejection. Both these modifications offer higher number of channel switching opportunities (due to reduced duration of OHS block) resulting in faster orthogonalization and hence, higher reward.

We first begin with the network potential which is an indication of the time required to reach SOC. The plots showing the variation of network potential with respect to time are shown in Fig.4(a) and Fig.4(c) for \( N = 5 \) and \( N = 10 \), respectively.
The decrease in the potential with time followed by constant potential shows that corresponding algorithm allows a network to reach SOC. However, proposed algorithms consistently offer faster decrease in potential and lower average potential than the CSM_MAB algorithm as highlighted in Fig. 4 (a) and Fig. 4 (c). Next, we consider average reward at different instants of the horizon. The average reward is the total reward of all user at a given time slot averaged over 100 independent experiments. As shown in Fig. 4 (b) and Fig. 4 (d), proposed algorithms offer higher reward (and hence, throughput) than the CSM_MAB algorithm. Significantly higher reward in the beginning ($t < 40000$) also indicates early orthogonalization to reach SOC which is an useful characteristic in the short horizon scenario such as dynamic networks. Constant potential and reward plots also indicate that the learning of channel statistics is accurate and proposed algorithms allow user to have sufficient samples of each channels thereby reducing switching to sub-optimal channels. These observations validate the Theorem 1.

One of the reasons behind the superior performance of the proposed algorithms is that they allow a higher number of channel switching opportunities as demonstrated in Fig. 5. Also, the difference between proposed and CSM_MAB algorithm increases with the increase in $N$. This is because the proposed algorithms allow each user to become master once in every OHS block while the CSM_MAB algorithm makes user compete for grabbing the channel switching opportunities. When multiple users compete, no one gets the opportunities leading to poor performance in spite of using complex radios with the wideband sensing capability. Among the dSOC_SN and dSOC_SN_H algorithms, dSOC_SN_H offers higher number of channel switching opportunistic in a given horizon due to reduced duration of the OHS phase.

Next, we consider large size ad-hoc network with $K = 50$ channels and $N$ ranging from 5 (sparse network) to 50 (dense network). In Fig. 6a, we compare the average reward of all users for different values of $N$. As expected, the reward increases with the increase in $N$ for all three algorithms. It can be observed that the proposed algorithms offer higher reward than the CSM_MAB algorithm for all $N$ and the difference increases with the increase in $N$ due to the same reasons discussed above.

Next, we analyze the difference between the dSOC_SN and dSOC_SN_H algorithms based on the number of collisions faced by each user throughout the horizon. Each collision leads to re-transmission of the lost packed leading to wastage of spectrum, time and power. Thus, they should be as small as possible. As shown in Fig. 6b, though dSOC_SN_H offers higher reward, it also leads to higher number of collisions.
The dSOC_SN algorithm offers approximately half the number of collisions than dSOC_SN_H due to longer OHS phase which means fewer switching opportunities as shown in Fig. 6. However, the number of collisions per user are less than 450 for a horizon size of 100000 which corresponds to very small collision probability of 0.005. Thus, proposed algorithms do not incur large number of collisions even though our signaling scheme for channel switching is based on collision. Next, we consider the dynamic networks.

![Graph showing average reward and number of collisions](image)

**Fig. 6:** (a) Average reward of all users, and (b) Total number of collisions faced by each user for different values of $N \in \{5, 10, ..., 50\}$ in the static ad-hoc network with $K = 50$.

### B. Dynamic Network

For dynamic networks, we consider three different scenarios depicting the various combinations of the time interval at which the users enter or leave the network. We mark the time of entry and exit of the user with a green and black dashed lines, respectively. We set $K = 10$ and each result shown here is the average of the values obtained over 100 independent experiments. The channel statistics are chosen randomly in each experiment.

In the first scenario shown in Fig. 7 (a) and (b), there is one user in the network at the beginning. New users enter into the network at $t = 25000$ and $t = 75000$. Also, single user leaves the network at $t = 50000$ and the leaving user is chosen at random from the set of the existing users. As expected, the reward performance of both algorithm is identical since the number of users are small which makes it easy to reach SOC. Note that network potential changes drastically whenever new user enters or leave the network. This is because, after every entry or exit event, the network may not be in SOC and needs finite time to come back to SOC.

Next we consider more challenging scenario with three users in the beginning. Thereafter, at every 10000 time slots, we alternate between user exiting and entering the network with leaving user chosen randomly. It can be observed from Fig. 7 (c) and (d) that the proposed algorithm offers significantly higher reward than the CSM_MAB_DN algorithm (extended version of the CSM_MAB algorithm using our synchronization scheme). This is expected as proposed algorithm has shown to outperform CSM_MAB for short horizon scenario in static networks. Also, the average network potential of the proposed algorithm is lower indicating faster orthogonalization to SOC and higher number of channel switching opportunities. Note that the network potential increases whenever a new user enters into the network and then decreases with time as network converges to SOC. However, when a user leaves the network, network potential decreases first and it may increase or decrease later depending on the channels vacated by leaving users. For instance, network potential may increase if the channel vacated by user is sub-optimal for one or more existing users and the users do not have sufficient samples of that channel. In such case, UCB algorithm forces users to explore that channel thereby leading to increase in the network potential. However, after learning the channel statistics, network comes back to SOC again.

We consider third scenario with more number users. In the beginning, there are five users and new users enter the network at $T = 20000, 30000$ and $50000$ whereas a user leaves the network at $T = 42000, 60000$ and $70000$. As shown in Fig. 7 (d) and (f), proposed algorithm offers higher reward and lower network potential than CSM_MAB_DN algorithm. From all three scenarios, we can observe that the difference between the performance of the two algorithm increases with the increase in the number of users, $N$.

### VII. Conclusions

In this paper, we presented distributed algorithms to achieve stable orthogonal configuration (SOC) in static as well as dynamic ad-hoc networks. We provided the detailed analysis of the proposed algorithms and validated their performance through simulated experiments for small as well as large size ad-hoc networks. The two novel contributions of the proposed algorithms are: 1) Need of low complex narrowband radio terminals compared to wideband radios in existing works, and 2) Epoch-less approach for dynamic networks. In addition, the proposed algorithms allow new users to synchronize in the network independently without the need of central controller or continuous sensing when the user is non-active. This feature might be useful for non-active users allowing them to use energy efficient sleep mode thereby increasing the lifetime of the battery operated radio terminals.

In future, we would like to extend the proposed algorithms for guaranteeing optimal orthogonal configuration to achieve higher throughput in addition to SOC. Another interesting scenarios include non-stationary channel statistics, delayed and complex feedback. Also, the proposed algorithms need user terminals to have sensing capability. For applications such as wireless sensor networks, it is preferable to have terminals without additional sensing hardware. The design of distributed algorithm for such terminals is extremely challenging and is one of the open research problems.

### References

[1] S. Parkvall, E. Dahlman, A. Furuskär, and M. Frenne, "NR: The New 5G Radio Access Technology," in IEEE Communications Standards Magazine, pp. 24-30, Dec. 2017.
Fig. 7: Average reward and average network potential comparison between dSOC_DN and CSM_MAB_DN algorithms for dynamic ad-hoc networks with three different scenarios.

[2] X. Hong, J. Wang, C-X. Wang and J. Shi, “Cognitive Radio in 5G: A Perspective on Energy-Spectral Efficiency Trade-off,” in IEEE Communication Magazine, vol. 52, no. 7, July 2014.

[3] M. Ozger, F. Alagoz, and O. Akan, “Clustering in Multi-Channel Cognitive Radio Ad Hoc and Sensor Networks,” in IEEE Communication Magazine, pp. 156-162, April 2018.

[4] A. Ali et. al, “Channel Clustering and QoS Level Identification Scheme for Multi-Channel Cognitive Radio Networks,” in IEEE Communication Magazine, pp. 164-171, April 2018.

[5] A. Anandkumar, N. Michael, A. Tang and A. Swami, “Distributed Algorithms for Learning and Cognitive Medium Access With Logarithmic Regret,” IEEE Journal on Selected Areas in Communications, vol. 29, no. 4, pp. 731–745, April 2011.

[6] Y. Gai, B. Krishnamachari, “Distributed Stochastic Online Learning Policies for Opportunistic Spectrum Access,” IEEE Transactions on Signal Processing, vol. 62, no. 23, Dec. 2014.

[7] M. Zandi, M. Dong and A. Grami, “Distributed Stochastic Learning and Adaptation to Primary Traffic for Dynamic Spectrum Access,” IEEE Trans. on Wireless Comms., vol. 15, no. 3, pp. 1675–1688, Mar. 2016.

[8] L. Besson and E. Kaufmann, “Multi-Player Bandits Models Revisited,” in Proceedings of the 29th International Conference on Algorithmic Learning Theory (ALT 2018), pp. 56–92, Lanzarote, Spain, 2018.

[9] G. Lugosi, and A. Mehrabian, “Multiplayer Bandits Without Observing Collision Information,” arXiv:1808.08416v1 [cs.LG], Aug. 2018.

[10] J. Bistritz, and A. Leshem, “Distributed Multi-Player Bandits - a Game of Thrones Approach,” arXiv:1810.11162v2 [cs.GT], Oct. 2018.

[11] O. Avner and S. Mannor, “Concurrent Bandit and Cognitive Radio Networks,” in Machine Learning and Knowledge Discovery in Database, pp. 66–81, Springer, April 2014.

[12] J. Rosenski, O. Shami and L. Szlak, “Multi-Player Bandits - a Musical Chairs Approach,” in Proceedings of 33rd International Conference on Machine Learning (ICML), pp. 1–9, New York, USA, 2016.

[13] Rohit Kumar, A. Yadav, S. J. Darak, and M. Hanawal, “Trekking Based Distributed Algorithm for Opportunistic Spectrum Access in Infrastructure-less Network,” in 16th Int. Symp. on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), May 2018.

[14] H. Joshi, R. Kumar, A. Yadav and S. J. Darak, “Distributed Algorithm for Dynamic Spectrum Access in Infrastructure-less Cognitive Radio Network, to appear in Proc. of IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain, 2018.

[15] M. Hanawal, and S. Darak, “Multi-Player Bandits: A Trekking Approach,” arXiv:arXiv:1809.00640v1 [cs.LG], Sept. 2018.

[16] G. Zhang, A. Huang, H. Shan, J. Wang, T. Q. S. Quek and Y. D. Yao, “Design and Analysis of Distributed Hopping-Based Channel Access in Multi-Channel Cognitive Radio Systems with Delay Constraints,” IEEE J. on Sel. Areas in Comms., vol. 32, no. 11, pp. 2026–2038, Nov. 2014.
Proof of Lemma 1

We want to compute $T_{rh}$ such that all users are on non-overlapping channels with high probability within $T_{rh}$. If $p_c$ denote the collision probability of a user when all the users are randomly hopping at any time slot $t$, and if none of the other users are on the non-overlapping channel (worst-case) then the probability that a user will find a non-overlapping channel within $T_{rh}$ is given by:

$$
\sum_{i=1}^{T_{rh}} p_c^{i-1}(1 - p_c).
$$

We want this probability to be at least $1 - \frac{\delta}{K}$ for each user. Hence we set

$$
\sum_{i=1}^{T_{rh}} p_c^{i-1}(1 - p_c) \geq 1 - \frac{\delta}{K}
\iff
1 - p_c^{T_{rh}} \geq 1 - \frac{\delta}{K}
\iff
T_{rh} \log p_c \leq \log \left( \frac{\delta}{K} \right)
\iff
T_{rh} \geq \frac{\log \left( \frac{\delta}{K} \right)}{\log p_c}.
$$

We next give a uniform upper bound on $p_c$. Note that in any round some users may be locked (call them locked users) while others selecting the channel uniformly at random (call them RH users). Fix a time slot $r$ and let $N_r \geq 1$ denote the number of users selecting channels uniformly at random. Let $p_{cr}$ denote the probability that collision is observed from an RH user. We have

$$
1 - p_c = P\{\text{no collision from RH users}\} + P\{\text{no collision from locked users}\}
\geq \sum_{j=1}^{N_r} \frac{(1 - p_{cr})}{K} \geq \frac{(1 - p_{cr})}{K}
\geq \frac{(1 - 1/K)^{N_r - 1}}{K} \geq 1/4K (\text{for all } K > 1).
$$

Theorem 1

Proof of Lemma 2

We want to compute $T_s$ such that new user enters into the SMCS phase and it is the sum of duration of three events: 1) Time required to enter the piggyback phase ($T_p$), 2) Time required to identify the CT/CS slots, index of the MB and beginning of the OHS block, ($T_i$) and 3) Time required to identify the reserved channel, ($T_r$).

When new user enters into the network, she sequentially senses the channel and enters into the piggyback phase if it is occupied. The worst case corresponds to the network with a single active user and it takes at most $2K$ time slots for new user to find the occupied channel. Thus, $T_p = 2K$.

After entering into the piggyback phase, the new user can identify all parameters immediately whenever old user leaves the network. This is because user can leave only when she is master. Else, the new user needs to sense the channel for at least $4K$ time slots. This is because the old user will be silent in the first SB of each MB when she is not a master and MB duration is $2K$ slots. Thus, the duration between consecutive silent SSBs is at the most $4K - 2$ slots. This is due to fact that user transmits in SSB when he is master and he can be master only once in OHS block. Thus new user can sense consecutive transmissions over $2(K - 1)$ slots before the old user becomes master and further consecutive transmissions over $2K$ slots when old user’s reserved channel is the most preferred channel. Thus, the new user has to wait for one more SB (i.e. SSB for subsequent MB) to identify all block parameters i.e. total $2(K - 1) + 2K + 2 = 4K$ slots. However, in the worse case, the user may have to wait for $K$ SSBs till old user becomes master and hence, he needs to sense the same channel for at most $2K^2$ slots. Then, $T_i = \max(4K, 2K^2) = 2K^2 \forall K > 1$.

Next, new user needs to identify the reserved channel before entering into the SMCS phase. As discussed before, the new user sequentially senses the channels and locks on the vacant channel. The worst case corresponds to $(K - 1)$ active users and master switches to the vacant channel. In this case, the new user needs to sense all channels except reserved channel of a master for at most $2K$ time slots before realizing the master switch and then occupying the master’s previous reserved channel in the next time slot. Thus, $T_r = 2K + 1$. Then,

$$
T_s = 2K + 2K^2 + 2K + 1 = K(2K + 4) + 1.
$$

This completes the proof.

Proof of Lemma 3

The proof of Lemma 3 is based on Theorem 1. When a new user enters into the network, she has to learn the statistics of remaining $(K-N)$ channels as rest of the channels are occupied by users and those users are not interested in these $(K-N)$ channels. Thus, using Theorem 1, we have

$$
t_m^{nu} \leq \frac{M - 1 - \sqrt{(M-1)^2 - 4M}}{2}, \text{ where } M := \frac{16(K-N)}{\Delta^2_{min}}.
$$

Thus, for all $t \geq t_m^{nu}$ every switch to one of the $(K-N)$ channels results in decrease in the network potential with probability $(1 - 2t^{-\alpha})$. Thereafter, new user needs only one opportunity to become master and switch to the most preferred channel among $(K-N)$ vacant channels. Using the Theorem 1, we have $T_s^{nu}(\delta) = t_m^{nu} + \log \left( \frac{\delta}{P_s^{nu}} \right)$ where $P_s^{nu} = 1 - 2(t_m^{nu})^{-\alpha}$. 

Proof of Lemma 4

When one of the $N$ users leaves the network, each of the remaining users needs at least $N - 1$ opportunities to become master and check the feasibility of channel swap or switch to the channel vacated by leaving user. Since the network was in SOC, Theorem 1 guarantees that all users have the sufficient number of samples of each channel and every swap or switch guarantees the decrease of network potential with high probability. Thus, the network will be in SOC again after $(N - 1)$ OHS blocks, i.e. $2K^2(N - 1)$ time slots. Since $N$ is unknown and $N \leq K$, the maximum number of slots required by the network to reach SOC is $T^{d} = 2K^2(K - 1)$. ■