This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2832021, IEEE Transactions on Network Science and Engineering

Hierarchical and Hybrid: Mobility-compatible Database-assisted Framework For Dynamic Spectrum Access

Ying Dai, Jie Wu, and Xiaojiang Du
Department of Computer and Information Sciences,
Temple University, Philadelphia, PA 19122
Email: {ying.dai, jiewu, dux}@temple.edu

Abstract—The protection of primary user activities plays a vital part in dynamic spectrum access. There are currently two types of schemes. One is based on spectrum sensing and the other one relies on spectrum database. There exist many works surrounding the subject of spectrum sensing, which requires high sensing accuracy, and thus poses extra time cost. Nevertheless, the database-based access scheme is receiving an increasing amount of devotion, e.g., IEEE 802.11af for TV white space. In comparison to spectrum sensing, nodes only need to look up the spectrum maps provided by the database. In this paper, we study the practical issues of the above two schemes, and propose a hierarchical framework, which enables the hybrid spectrum access scheme. In this framework, we build relatively reliable clusters, and have cluster heads connect to the spectrum database to assist nodes in their clusters. As a result, nodes with poor or no connections to the database can benefit from spectrum maps as well. The process of retrieving spectrum maps is formalized as a Markov Decision Process. Moreover, mobility compatibility is provided. We illustrate how the evolution of our hierarchical structure is affected by mobile nodes. We also propose a feasible algorithm to maximize the benefits that can be obtained from the database under the mobile environment. We model a virtual database and conduct simulations to reveal certain performances of our framework.

Index Terms—*Dynamic spectrum access, database-assisted, mobility, hierarchical, hybrid.*

1 Introduction

Recently, the spectrum congestion problem has attracted a growing amount of attention. Dynamic spectrum access[1–3] is a promising solution, which enables spectrum sharing between primary users (PUs) and secondary users (SUs), while protecting the PU activities from being interfered with by SUs. The key issue lies in the protection of PUs, which requires each SU, or node, to make sure that no active PU exists when accessing the spectrum.

Many current researches focus on improving the accuracy of spectrum sensing techniques (e.g.,[4–7]). That is, each node senses PU activities on the spectrum before accessing it, and will only use the spectrum when it senses no active PUs. However, in order to avoid interfering with PUs, a SU node needs to perform spectrum sensing very frequently. This makes sensing time become a large portion in the overall spectrum access time cost.

In the TV white space spectrum, there exists a different framework to implement spectrum access, i.e., IEEE 802.11af[8], which is a standard for the TV white space spectrum sharing. The framework is a database-based scheme, which provides an architecture to help each node to gain the spectrum map at its current location. Based on the spectrum map, the node is aware of the spectrum availabilities, i.e., when PUs are not using the spectrum. Hence, spectrum access becomes efficient for each node, since it only needs to look up the corresponding spectrum map, rather than frequently performing spectrum sensing.

The database-based, or spectrum map-based, scheme ([9–11]) sheds light on the dynamic spectrum access problem in general, rather than the white space only. Aside from the extendability of constructing such spectrum databases, there are some other issues that need to be considered, even in the white space. Firstly, the reachability of the database is questionable. A node needs to have 3G, Wifi, or a similar environment of building a connection with the database, either directly or through some agent. Nevertheless, this is unrealistic for all nodes considering their varieties. Some nodes may be in better condition to connect to the database, and the interference would increase if all nodes try to build connections to the database. The second issue is that although spectrum maps effectively reduce the time cost by avoiding spectrum sensing, the time cost spent on sessions between nodes and the database has to be taken into account, especially when nodes are mobile.

Therefore, a hybrid scheme of the spectrum map-based and sensing-based approaches is inevitably proposed. Yet, this does not mean that a node should switch to spectrum sensing immediately if the database becomes unreachable. To better utilize the database, when a node fails to connect with the database, it might gain assistance from its neighbors, which are one-hop away and are easy to reach. In Fig. 1, we divide the area into grids based on spectrum availabilities. Each grid has its own spectrum map, which means the spectrum availability remains stays the same

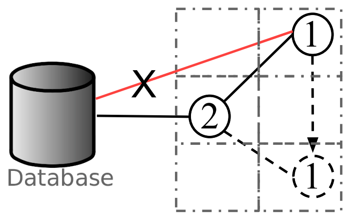


Fig. 1. Node 1 can gain assistance from node 2. The movement of node 1 causes extra costs. Each grid is associated with a spectrum map.

within one grid. For simplicity, squares are used to depict grids, although grids are not always necessarily squared. Suppose that node 1 cannot reach the database. However, it may access the database with the assistance of its neighbor, node 2, which can fetch the spectrum map for node 1. Consequently, a general structure that enables the assistance for nodes during database access can be beneficial.

In addition, when nodes are mobile, the problem becomes more complicated, considering the cost of fetching spectrum maps. For example, in Fig. 1, if node 1 moves to a new location, it needs to request the spectrum map again for the new grid. The frequent sessions between nodes and the spectrum database pose potential conflicts among nodes, and increase the communication cost. Fortunately, the idea of the speculative execution technique has the potential to help reduce the communication cost. The main idea is to do work in advance, before knowing whether that work will be needed. This is helpful in preventing the delay of doing the work after knowing the necessity. A similar theory can be applied for fetching spectrum maps by the mobile nodes. When a moving node sends requests for spectrum maps, it fetches the spectrum maps for its future locations. Then, the frequency of connecting to the database is reduced.

Motivated by the above discussions, in our paper, we provide a hierarchical spectrum access framework. Under such a hierarchical structure, each node adopts the hybrid scheme of both spectrum map-based and sensing-based approaches. We construct relatively reliable clusters, and have cluster heads to connect to the spectrum database to assist the nodes in their clusters. The cluster heads are selected based on their relative mobility metrics, storage limits, and connection qualities to the database. The connections between the databases and nodes are divided into two layers: from the database to the cluster heads, and from the cluster heads to other nodes. In this way, the communication cost and potential conflicts associated with reaching the database are reduced.

Moreover, to fully maximize the benefits for each cluster, we formalize the process of cluster heads retrieving information from the spectrum database as a Markov Decision Process (MDP). Each cluster head can determine the best actions regarding whether to connect to the spectrum database, or which spectrum maps to fetch, e.g., the spectrum maps for future locations. We further study the mobility influences on the evolution of our hierarchical structure. Afterwards, a practical algorithm based on the mobility factors for the MDP on each cluster head is proposed.

The main contributions in our work can be described in the following aspects:

- We propose a hierarchical framework enabling the hybrid spectrum access scheme, which increases the utilization of the spectrum database and reduces the overall cost.
- We select the cluster heads based on a novel metric, which considers the relative mobility metric, the storage limit, and the quality of the connection to the database.
- We formalize the decision process at each cluster head as an MDP, which reduces the average spectrum access time for each cluster.
- We resolve the mobility influences by providing the evolutions of our hierarchical structure and by proposing a feasible solution in the mobile environment.

Our paper is organized and proceeds in the following order. In Section 2, we discuss the related works. The problem formulation is shown in Section 3. Our spectrum access framework is presented in detail in Section 4, which contains the construction of our hierarchical structure, the MDP formulation, and the study of the mobility influences. The performance evaluation is described in Section 5. We conclude our paper in Section 6.

2 Related Works

In this section, we introduce the related works from two aspects. The first one is the database-driven spectrum access scheme. The second one is about the cluster applications for spectrum access.

2.1 Database-driven Spectrum Access

In [12], the authors present SenseLess, which is a database-driven white space network. Their system relies fully on a database service to access white space spectrum, rather than spectrum sensing. It provides a complete service to ensure an efficient white space networks while protecting PUs. A game theoretic approach is proposed in [13] for the database-assisted white space access point network design. Authors in [14] build the radio environment map for the realization of dynamic spectrum access. Their model makes use of multi-domain information from geolocation databases, e.g., characteristics of spectrum use, geographical terrain models, propagation environment, and regulations. They model the channel selection problem as a distributed game in each access point and prove the convergence of a state-based Nash equilibrium. The database information itself is discussed in [15]. They focus on improving the accuracy of the geolocation databases through their collected spectrum sensing samples in a TV network area. The work in [16] compares different estimation methods and studies the accuracy of the signal strength estimation for a primary TV network, which is used for including measurement data into a database's prediction process. The location robustness and privacy issues in the database-driven networks are studied in [17] and [18]. Authors in [19] focus on the optimization under the database-driven model. Authors in [20] consider the application perspective, and propose the quality-of-experience metrics, for achieving a better service based on a spatio-temporal geo-database. Our work considers the practical issues where not all nodes can connect with the database, and enables the hybrid spectrum access scheme, as opposed to the database-driven scheme, which is only based on spectrum maps.

2.2 Cluster Applications for Spectrum Access

The model in [21] relies on the geographical location information and proposes a virtual backbone construction scheme in cognitive radio networks without a control channel. Cluster structures are applied in [22] for the allocation of control channels for spectrum access. Different channels for control are allocated at various clusters in the network, and the problem is solved by the bipartite graph. An event-driven cluster construction scheme is proposed in [23] for cognitive radio sensor networks. When an event happens, to better deliver the message, the detection of the event forces the eligible nodes in the network to form into clusters. Different from the above works, our clusters are used for the hierarchical structure for spectrum access. The cluster heads are selected based on our proposed weight definition. We also consider the evolution of the clusters under the mobility influences. Several literature (e.g., [24–26]) has studied the related spectrum sharing issue.

3 Problem Formulation

In this section, we first describe the network environment for dynamic spectrum access. Then, we formulate the system objective and constraints.

3.1 Network Environment

We consider that in a dynamic spectrum access network, there is a set of PUs that have privileged use on the spectrum. The SUs, or nodes, are mobile and make opportunistic use of the spectrum while protecting the PU activities. We assume that there is a common control channel (CCC) for nodes to connect to the database and exchange control messages. Suppose that there is an accurate spectrum database to provide spectrum maps at different locations, e.g., the geolocation database which provides the white space map. A spectrum map contains the PU activities and shows the spectrum availability at a location. Since spectrum availabilities are usually similar in nearby locations, we assume that the network is divided into different grids, and each grid has the same spectrum availability. The spectrum maps are stored in the database with the grid IDs as their "primary keys". For example, one entry denotes the spectrum map in one grid. Therefore, the spectrum map is likely to change when a node moves to a new location, that is in a new grid. Another assumption is that nodes will not behave abnormally, for example, launching DNS attach. Network security is not a focus in this model.

The nodes can retrieve spectrum maps, by sending their current locations to the database. The database returns the corresponding spectrum maps based on the grid IDs of their locations. The connections between nodes and the database are wireless connections, e.g., through 3G or Wifi. It is likely that some nodes are unable to download spectrum maps from the database due to connection failures. Thus, the following two spectrum access schemes with different time costs are both needed:

1) Spectrum map-based: Nodes with the spectrum maps returned by the database can access the spectrum simply based on the information of the spectrum map. The time cost of looking up the spectrum map to find the available part is denoted as TD;
2) Sensing-based: Nodes without the spectrum map must sense the spectrum first and access the portion that has no active PUs. The time cost of sensing until finding the available spectrum is denoted as TS.

We assume that $TD < TS$, since the time cost of querying the local spectrum map depends on the communication delay, while spectrum sensing is a continuous process and is required at the beginning of each time slot. As a result, if there is no extra cost besides TD, a node would favor the spectrum map-based access scheme over the sensing-based scheme.

In fact, the communication between a node and the database takes extra costs. It consists of two parts: 1) constructing the session between the node and the database; 2) downloading the spectrum map from the database. We use TC to denote the time cost for session construction. The time cost for downloading is related to the size of spectrum maps. For simplicity, we use λTM to denote the time cost for the downloading phase, where TM is the constant time cost for downloading a spectrum map of a single grid and λ is the number of spectrum maps that are fetched from the database. Hence, the expected time cost for a node u to access the spectrum is:

$$T_u = p \times (TD + TC + \lambda TM) + (1 - p) \times TS, \tag{1}$$

where p is the percentage that node u adopts the spectrum map-based scheme.

3.2 System Objective

Our objective is to minimize the overall T_u for all nodes in the network. Here, we do not include the time cost due to conflicts after an available spectrum is chosen, which is out of the scope of this paper. The first constraint is to protect the PUs from the nodes' interference. Another constraint is that the storage size is limited at each node. Otherwise, a node could download all the information at the database, which is obviously unrealistic. We use E_u to denote the maximum number of spectrum maps that can be stored on node u. Therefore, $\lambda \leq E_u$.

Also, due to the mobility of nodes in the network, it poses two main challenges to our model:

1) Since the spectrum availability varies at different grids, a node may need to download the new spectrum map when it moves to another location. The frequent connections to the database bring larger values of TC and λ in Eq. (1).
2) It is likely that a node moves to a location with poor or no connections, and the database becomes unreachable. As a result, a node has no spectrum map and must perform spectrum sensing at the time cost TS.

With the above factors taken into consideration, it is implausible to provide an optimal solution to minimize the overall spectrum access time cost in the network. In the next sections, we will discuss our hierarchical framework with the hybrid spectrum access scheme, which considers the above issues and supports mobility well.

4 Spectrum Access Framework

In this section, we will describe the overview of our framework. Then, we will introduce each phase in detail.

4.1 Framework Overview

Our spectrum framework is mainly based on two motivations. First, considering the benefits brought by spectrum maps, our framework should facilitate nodes without reliable connections to the database. Second, the mobility issue must be taken into account by our framework. The overview of our framework is as follows:

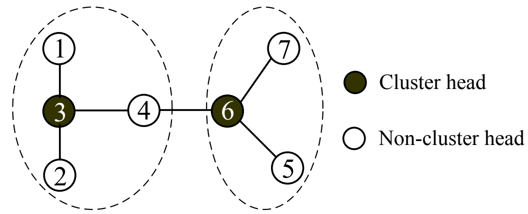
- Select cluster heads among the nodes and construct clusters. Each node is aware of its cluster head. The hierarchical structure is built upon the clusters. A hybrid spectrum access scheme is adopted under this structure.
- The cluster heads connect to the database and retrieve spectrum maps for nodes in their clusters. This process of each cluster head needs to be optimized and is formulated as a MDP.
- Considering the influences of node mobilities, an adaptive and practical scheme is applied for the evolution of the hierarchical structure and spectrum access.

For nodes that have no reliable connections to the database, they can still avoid spectrum sensing by fetching spectrum maps from the cluster heads. In other words, the sessions between the nodes and database are delegated to a subset of the total nodes, which contains only the cluster heads. With fewer nodes connecting to the database, the session construction cost TC in Eq. (1) is reduced. In the next subsections, we will discuss our framework in detail from the following aspects: 1) How the cluster structure is constructed for hybrid access; 2) The overview and challenges of sessions between secondary nodes and the spectrum database; 3) How each cluster head makes the decision about building its session with the database; 4) The adaptive scheme for each node when taking possible mobility effects into account.

4.2 Hierarchical Structure Construction For Hybrid Access

To build the hierarchical structure, we first need to construct the clusters by defining a weight for each node as a basis for cluster head selection. Different from traditional approaches, e.g., the node IDs, the relative distances, and so on, the weight definition here has some different factors that we need to consider.

Intuitively, we tend to choose nodes that are less mobile, compared to their neighbors. Moreover, the cluster heads should have stable connections to reach the database. Last but not least, the cluster heads need to have more available storage space for storing the downloaded spectrum maps from the database. Considering these three factors, we give the definition of the node weight for cluster head selection:



ID	1	2	3	4	5	6	7
E_u	2	3	4	2	3	2	2
M_u	2.6	1.1	0.5	1.3	3.6	0.3	1.2
a	0.6	0.8	0.8	0.7	0.7	0.9	0.9

Fig. 2. An example of cluster head selection by calculating of each node's weight based on Definition 1.

Definition 1. ***Weight for Cluster Head Selection.*** *Given a node u, its weight W_u for cluster head selection basis is:*

$$W_u = \alpha \frac{E_u}{M_u},$$

where α is the ratio of the successful connections to the database, E_u is the maximum number of spectrum maps that can be stored on node u, and M_u is the aggregate local mobility value at node u.

The value of α can be calculated from the historical data. E_u is easily known by node u itself, which is proportional to its current available storage size. There have been many works [27] defining the local mobility metrics. We calculate the value of M_u as:

$$M_u = var[d_{uv}], \forall v \in N_u,$$

where N_u is the neighbor set of node u, d_{uv} is the distance difference between u and v in a constant sample time duration, and var is the variance of all d_{uv} with respect to 0. M_u can be obtained through two successive message exchanges from each neighbor on CCC, and the calculation of the pairwise relative distance difference d_{uv}. A higher value of M_u indicates that u is more relatively mobile, compared to nodes in N_u.

Having the weight defined, the cluster construction can be performed based on the classical existing scheme with 2-hop information [28]:

1) All nodes gather the weight values of 2-hop nodes over the CCC and mark themselves as uncovered;
2) An uncovered node u becomes a cluster head, if it has the highest weight W_u among all its neighbors in N_u;
3) The selected cluster heads and their connected neighbors are marked as covered;
4) Repeat Steps 2 and 3 until all nodes are covered.

The coverage and connectivity has been proved in [28]. If two nodes have the same weight, the one with the higher

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2832021, IEEE Transactions on Network Science and Engineering

node ID is chosen. Each node belongs to one cluster, which means there is no overlap between clusters. One simple example is shown in Fig. 2. With the values defined in the table, the weight of each node can be calculated. For example, $W_3 = 4/0.5 \times 0.8 = 6.4$ and $W_6 = 2/0.3 \times 0.9 = 6$. The selected cluster heads are nodes 3 and 6. Among the remaining nodes, nodes 1, 2, and 4 select 3 as their cluster head. Nodes 5 and 7 select 6 as their cluster head. In addition, the cluster size constraint can be added. For example, each cluster has a threshold on its size limit. Our focus here is to provide the hierarchical framework for hybrid spectrum access, and will leave the size constraints for future work. Moreover, even though the selected cluster heads are less mobile with respect to the other nodes, it is inevitable that the clusters need to change, e.g., the cluster heads move to meet each other, nodes move from one cluster to another, and so on. We will discuss the corresponding adjustments later.

After the clusters are constructed, node u will access the spectrum based on the following process:

1) When u needs to access the spectrum and has no knowledge of the spectrum map, it sends a request containing its location information to the cluster head;
2) After the cluster head receives the request, it checks whether its current storage contains the spectrum map of u's location;
3) If the spectrum map exists at the cluster head's storage, the cluster head will send the map to u. Otherwise, it replies with no available spectrum map;
4) Node u adopts the hybrid spectrum access scheme. If u receives the spectrum map, it will access the spectrum based on the map. Otherwise, it will switch to the sensing-based scheme.

4.3 Markov Decision Process For Session With Database

After the hierarchical structure is constructed, the most important task is for each cluster head to download the spectrum maps from the database and provide corresponding assistances to nodes in the cluster. Since nodes are mobile and the spectrum availabilities vary at different grids, the spectrum maps stored at the cluster heads have to be updated. Each cluster head faces two questions regarding the session with the database:

1) When should it connect to the database and download the spectrum maps?
2) What are the most useful spectrum maps to download for helping nodes in the cluster?

For the first question, cluster heads need to construct the session with the database and download new spectrum maps when the nodes' located grids are changed. However, if a cluster head connects to the database every time it receives a new spectrum map request, the sessions with the database would become too frequent. As a result, it would pose more time cost during the frequent session constructions, and also cause more conflicts on the CCC.

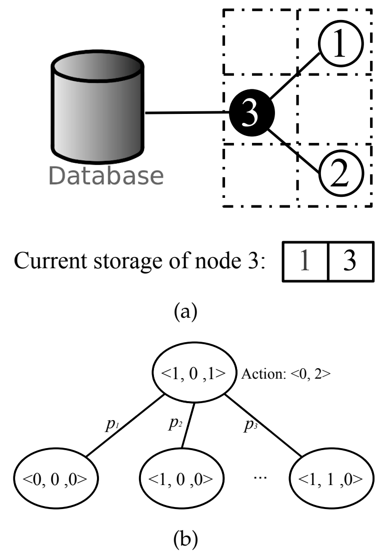


Fig. 3. The left side is the current storage on the cluster head. One example is that the current state is $< 1, 0, 1 >$. After taking action $< 0, 2 >$, the possible new state is $< 1, 1, 0 >$. The right side is an example of the state transition graph after the cluster head takes action $< 0, 2 >$.

For the second question, since nodes are mobile, a cluster head cannot simply download spectrum maps at the nodes' current locations, but also has to consider their possible locations later. This scenario is similar to the speculative execution. Cluster heads need to make decisions in advance about which spectrum maps are the most useful to fetch.

To answer the above two questions, we formulate the decision process on each cluster head as an MDP in the following parts. We use h to denote a cluster head, and $Cluster(h)$ to denote the set of nodes in the cluster led by cluster head h, including h itself. For example, in Fig. 2, $Cluster(3) = \{1, 2, 3, 4\}$. We use I_u, where $u \in Cluster(h)$, to denote whether u's spectrum map at its current location is available at h. Hence, I_h is a binary value, 0 or 1. We define the state set as follows:

Definition 2. ***State Set.*** *The state set of cluster head h is $S_h = \{s_h | s_h =< I_1, I_2, ..., I_{\mathcal{N}} >\}$, where $\mathcal{N} = |Cluster(h)|$, which denotes the total number of nodes in $Cluster(h)$.*

Obviously, the number of states is finite, with the maximum possible states equaling $2^{\mathcal{N}}$. Therefore, each state of h is a multi-tuple with binary values. For example, in Fig. 3(a), node 3 is the cluster head. Suppose $E_3 = 2$ (As explained before, E_3 is the maximum number of spectrum maps that can be stored on node 3). The set of nodes in $Cluster(3)$ is $\{1, 2, 3\}$. The current storage of node 3 stores nodes 1 and 3's spectrum maps of the grids corresponding to their current locations. Then, the state of node 3 is $< 1, 0, 1 >$, which means the spectrum maps at nodes 1 and 3's locations are in the cluster head's storage, while the spectrum map at node 2's location is not.

Definition 3. ***Action Set.*** *The action set of cluster head h is $A_h = \{a_h | a_h =< R_1, R_2, ..., R_{E_h} >\}$. $R_i = 0$ if the ith entry in h's storage is not replaced. If the ith entry in h's storage*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2832021, IEEE Transactions on Network Science and Engineering

is replaced with the spectrum map at the grid with ID g, then $R_i = g$, where $i \in [1, E_h]$, and $g \geq 1$.

Here, the action set is also finite, with the maximal size equaling $(\mathcal{N}\beta + 1)^{E_h}$, where β is the expected number of the possible grids that a node can be at in the next time slot, and its value depends on the mobility model. A cluster head takes an action at the beginning of each time slot. This does not mean that the cluster head needs to build connections with the database every time. For example, the cluster head does not need to do anything if it takes an action a_h where all the R_i are 0. To better illustrate the action set, we take Fig. 3(a) as a simple example. Again, node 3 is the cluster head. $E_3 = 2$, which means there are two entries at the cluster head. Suppose node 3 knows that the next position of node 2 is at the grid whose ID is 2. If the action $< 0, 2 >$ is taken, it indicates that node 3 does not replace the 1st entry in its storage, but replaces the 2nd entry with the spectrum map at grid 2. Then, the state in the next time slot becomes $< 1, 1, 0 >$.

Each action taken by the cluster head reflects different time cost T_h in Eq. (1) with u replaced by h. For an action a_h, if $\exists R_i \neq 0, \forall i \in [1, E_h]$, it means that the cluster head needs to connect to the database, and the value of TC increases at the cluster head. A larger number of nonzero R_i in a_h indicates a larger value of λ in T_h.

Having the action set defined, the reward function should take the time cost of the hybrid spectrum access scheme on each node in the cluster into consideration. Based on the definition of T_u in Section 3, we use $T(s_h)$ to denote the average time cost for nodes at state s_h in $Cluster(h)$:

$$T(s_h) = \frac{\sum_{u \in Cluster(h)} T_u}{\mathcal{N}}. \tag{2}$$

Since only the cluster heads connect to the database, the session construction TC and spectrum downloading time cost TM in Eq. (1) are 0 for nodes that are not cluster heads. Consequently, we define the reward as:

Definition 4. ***Reward.*** *For cluster head h, whose current state is s_h, the reward of taking the action a_h is defined based on the expected average spectrum access time cost of the cluster:*

$$U(s_h, a_h) = \sum_{s'_h \in S_h} P^{a_h}_{s_h, s'_h} \frac{1}{T(s'_h)}, \tag{3}$$

where s_h is the current state, s'_h is the new state, and $P^{a_h}_{s_h, s'_h}$ is the transition probability, which is the probability that taking action a_h in state s_h will lead to s'_h in the next time slot.

The larger value of the reward indicates a smaller value of the expected average time cost during spectrum access.

In Fig. 3(a), we assume that the cluster head knows the next position of node 2. Nevertheless, it is impractical for cluster heads to be aware of the exact future positions of other nodes. Fig. 3(b) demonstrates a more realistic state transition graph for Fig. 3(a) after node 3 takes action $< 0, 2 >$. Since nodes are mobile, it is likely that the information stored on the cluster head becomes useless or contains the spectrum maps for other nodes. Therefore, multiple states are possible to achieve. We need the transition probability to describe this process. Clearly, the value of the transition probability depends on the moving pattern of each node in the cluster, i.e., we apply the random walk model[29] in the simulation. The cluster head h is able to predict the next possible positions of nodes in $Cluster(h)$. Then, h can determine the value of $P^{a_h}_{s_h, s'_h}$ by checking whether the spectrum maps for the next possible positions of each node are at h, and deriving the probabilities of all the possible s'_h.

A cluster head needs to choose an action at the start of each time slot based on its current state. If we determine the best set of actions that a cluster head should take, obviously, the two questions raised at the beginning of this subsection can be answered. Therefore, we study the optimal stationary policy first and give a heuristic scheme later.

4.4 Optimal Stationary Policy

To begin with, we define the policy for a cluster head h:

Definition 5. ***Stationary Policy.*** *The stationary policy at cluster head h is denoted as $\pi \in A_h^{|S_h|}$, which maps from each state s_h to an action a_h, e.g., $\pi(s_h) = a_h$ is the action taken when the state is s_h.*

Suppose that the cluster head h's initial state is s_h. Having the stationary policy π, we can define the value function as the average time cost of the cluster:

$$V(\pi, s_h) = \lim_{T \to \infty} \frac{1}{T} \left[\sum_{t=0}^{T-1} U(s_h, \pi(s_h)) \right], \tag{4}$$

where T is the total number of time slots. The ideal solution is to find an optimal policy among all π to maximize $V(\pi, s_h)$. Next, we prove the existence of the optimal stationary policy.

Theorem 1. *There exists an optimal stationary policy for the MDP on each cluster head.*

Proof. The state space at each cluster head is finite, as explained above. We need to prove that every stationary policy leads to an irreducible Markov chain. For all $s_h \in S_h$, the transition probability $P^{a_h}_{s_h, s'_h}$ is greater than 0, since the storage on h is able to be updated to match each state. Thus, any two states can communicate with each other. Considering that the state space is limited, the resulting Markov chain is irreducible by way of any policy. Therefore, the optimal stationary policy exists on each cluster head. □

We use π^* to denote the optimal stationary policy. Because of the irreducibility of the MDP on each cluster head, the optimal stationary policy is independent from the initial state, i.e., $V(\pi^*) = V(\pi^*, s_h)$, $\forall s_h \in S_h$. $V(\pi^*)$ is the maximum value of the reward function. It is generally difficult to obtain the structure of the optimal policy. In addition, the complexity of achieving the optimal policy is very high. In the following parts, we focus on studying the mobility influences on the evolutions of the hierarchical structure and the MDP process at each cluster head, and proposing a heuristic solution.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2832021, IEEE Transactions on Network Science and Engineering

4.5 Adaptive Scheme With Mobility Influences

Although our cluster head selection takes into account the relative mobility metrics, it is still inevitable that the clusters need to be updated. Once the clusters are dynamic, it means that nodes are likely to move among ranges of different cluster heads. These dynamics not only cause the evolutions of our hierarchical structure, but also motivate our feasible solution.

4.5.1 Hierarchical Structure Evolutions

First, we study the cluster dynamics, which affect our hierarchical structure. It mainly includes two scenarios: 1) a mobile node moves out of the range of its original cluster head; 2) two cluster heads meet each other.

Scenario 1. Node u can determine whether it is out of the cluster head's cover, by sending the request to its cluster head. Here, u is not a cluster head, itself; otherwise, we categorize it to the second scenario. If it receives no feedback after three retries, u can decide if the first scenario happens.

After u concludes that it is no longer covered by the original cluster head, it calculates its current weight and broadcasts to its neighbors. For a neighbor v of u, $v \in N_u$, the behavior of v after receiving u's weight depends on whether v is a cluster head or not:

- If v is not a cluster head, it records u's weight, W_u and does not reply;
- If v itself is a cluster head and $W_v > W_u$, it replies its weight W_v to u.

If u receives no replies, which means there is no cluster head around, u claims itself as a cluster head and broadcasts the message. Any node v in N_u compares u's weight, which is recorded previously, with its current cluster head. If W_u is greater than the weight of v's current cluster head, v will join u's cluster. If u receives replies from other cluster heads, it will join the one with the maximum weight.

Scenario 2. For cluster head h, it can determine whether it meets another cluster head by overhearing the feedbacks from other cluster heads over CCC. If h overhears a message that contains the spectrum map information or claims that the requested spectrum map is unavailable, it will obtain the conclusion that another cluster head is within its range.

If the cluster head h concludes that it meets another cluster head, it will broadcast its weight along with its identity as a cluster head. If a cluster head h' receives the weight of h, h' replies with its own weight $W_{h'}$. Both h and h' compare their weights. The one with the lower weight will join the cluster of the one with higher weight. Suppose $W_h < W_{h'}$ and h joins $Cluster(h')$. Then, h broadcasts a resigning message. Nodes in $Cluster(h)$ will mark themselves as in Scenario 1, which indicates that they are no longer covered by any cluster head, and react according to the above scheme for Scenario 1.

With the adjustments under both scenarios, the following theorem stands:

Theorem 2. *For cluster head h, its weight is greater than or equal to any node in its cluster, i.e.,* $W_h \geq W_u$, $\forall u \in Cluster(h)$.

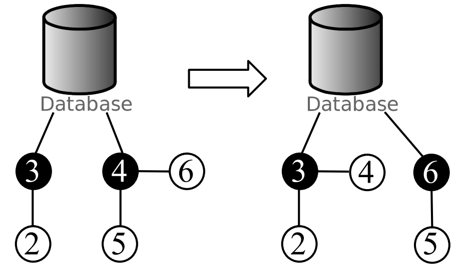


Fig. 4. An example of hierarchical structure evolution.

Proof. We use induction to prove this theorem. The base case is the initial clusters without any evolution in either of the two scenarios. Obviously, $W_h \geq W_u$, $\forall u \in Cluster(h)$ stands since it is how the cluster heads are selected.

The inductive hypothesis is that after x ($x > 0$) steps of the adjustments based on Scenario 1 or 2, $W_h \geq W_u$, $\forall u \in Cluster(h)$ is true.

If the $(x+1)$th step is the evolution under Scenario 1 and node u finds itself not covered by any cluster head, there are two possibilities. If u joins another cluster, then W_u must be less than or equal to the weight of its new cluster head. If u becomes a cluster head itself, then only nodes with lower weights than W_u will join $Cluster(u)$. If the $(x+1)$th step is the evolution under Scenario 2 and cluster head h needs to resign itself, then h will join a cluster with a larger weight of the cluster head. The nodes in the original $Cluster(h)$ are the same as node u in Scenario 1. Thus, the theorem is proven. □

The motivation behind our evolution scheme is to keep the dynamics local and the adjustments distributed. Under both scenarios, each node adopts the hybrid spectrum scheme based on the updated hierarchical structure. An example is shown in Fig. 4. On the left side, nodes 3 and 4 are the cluster heads, which fetch the spectrum information to assist nodes in their clusters, i.e., $Cluster(3) = \{2, 3\}$, $Cluster(4) = \{4, 5, 6\}$. Suppose that when Scenario 2 happens, cluster head 4 needs to resign itself and join $Cluster(3)$. Then, nodes 5 and 6 face Scenario 1. Node 6 becomes a new cluster head with $Cluster(6) = \{5, 6\}$. The right side of Fig. 4 gives the new hierarchical structure for spectrum access.

4.5.2 Practical Policy Generation For Cluster Heads

Due to the mobility influences on the stability of the hierarchical structure and the adjustments of clusters, the value of T in Eq. (4) has a upper bound. When the clusters are updated, the MDP on the new cluster heads also need to be reconstructed. Therefore, we use K to denote the expected number of time slots in which a cluster stays stable. The value of K depends on the mobility factors of nodes in the network, and we will study the influences of different K values in the simulation.

With the upper limit of the stable time slots, we can solve the decision process at each cluster head using a modified value iteration algorithm:

- First, we break up the process by the number of remaining steps. The total number of steps equals

Algorithm 1 Optimal policy generation for cluster head h with total K steps.

```
for each state s_h ∈ S_h do
    Q_0(s_h) = 0;
for k = 1 to K do
    for each state s_h ∈ S_h do
        for each action a_h ∈ A_h do
            Calculate Q_k(s_h, a_h) = U(s_h, a_h) + Σ_{s'_h ∈ S_h} P^{a_h}_{s_h,s'_h} Q_{k-1}(s'_h);
        Calculate π*_k(s_h) = arg max Q_k(s_h, a_h);
        Calculate Q_k(s_h) = Σ_{∀a_h ∈ π*_k} Q_k(s_h, a_h);
return π* = π*_K.
```

K since the cluster head needs to take an action at the start of each time slot.

- Second, given the optimal policy for $(k-1)$ steps to go, where $k \geq 1$, we compute the optimal policy for k steps to go, until k reaches K.

For a cluster head h, we use $Q_k(s_h)$ to denote the value of state s_h with k steps to go, and $Q_k(s_h, a_h)$ to denote the value of taking action a_h in state s_h with k steps to go. We set the initial vale $Q_0(s_h)$ as 0 for all s_h, which means there are no steps remaining at state s_h. We compute the values of each state from the values on the next step in a backward way:

$$\pi^*_k(s_h) = \text{arg max } Q_k(s_h, a_h)$$

$$Q_k(s_h) = \sum_{\forall a_h \in \pi^*_k} Q_k(s_h, a_h)$$

$$Q_k(s_h, a_h) = U(s_h, a_h) + \sum_{s'_h \in S_h} P^{a_h}_{s_h, s'_h} Q_{k-1}(s'_h),$$

where $\pi^*_k(s_h)$ is the optimal action that maximizes $Q_k(s_h, a_h)$ with k steps to go. The details of our algorithm on each cluster head are described in Algorithm 1. The setting of the value K reduces the complexity of our algorithm. Starting from 0 steps remaining at each state, Algorithm 1 calculates the optimal policy with K steps to go. For example, suppose $K = 3$. Based on Algorithm 1, the cluster head first initiates $Q_0(s_h) = 0$ for all s_h. Next, $k = 1$, the cluster head calculates $Q_1(s_h)$ and stores the action that maximizes $Q_1(s_h)$ for every s_h. Similarly, the actions that maximize $Q_2(s_h)$ and $Q_3(s_h)$ are stored for each state. Eventually, each state is associated with its optimal policy regarding $K = 3$. Next, we give the complexity analysis of Algorithm 1.

Theorem 3. *The complexity of generating optimal policy for a cluster head with total K steps is $O(2^{\mathcal{N}}(\mathcal{N}\beta + 1)^{E_h} Kl)$, where l is the maximum number of non-zero outgoing transitions.*

Proof. The number of $Q_k(s_h)$ values at each step is the number of actions times the number of states, which is $O(2^{\mathcal{N}}(\mathcal{N}\beta + 1)^{E_h})$, as explained in Section 4.3. The time cost for computing $Q_k(s_h)$ is $O(1)$. The total number of loops to calculate $Q_k(s_h)$ is $O(Kl)$, where l is the max number of non-zero outgoing transitions. Therefore, the total complexity of Algorithm 1 is $O(2^{\mathcal{N}}(\mathcal{N}\beta+1)^{E_h} Kl)$. □

TABLE 1
Simulation settings.

Parameters	Values
Number of nodes	$[20, 55]$
Transmission range of each node	$[5, 80]m$
Maximum speed	$[5, 15]m/s$
Pause times	$[0, 30]s$
TS	$0.5s$ per time
TD, TC, TM	$0.1s$, $0.5s$, $0.5s$
E_h	$[2, 6]$
K	$[2, 6]$
Number of PUs	50

Considering that the cluster sizes and values of K are relatively small among the mobile nodes, Algorithm 1 provides a practical solution for cluster heads to select optimal policy in a short term under the dynamic environment. We will study the influences of varying cluster sizes and K values in the simulations.

5 Performance Evaluation

In this section, we first describe our simulation settings. Then, we present the simulation results of our framework from different aspects.

5.1 Simulation Settings

Table 1 shows the overview of the main simulation parameters. We discuss the detailed settings from three aspects, SU model, spectrum map model, and evaluation metrics, in the following parts.

5.1.1 SU model

We randomly distribute a number of nodes in a network with range $200m \times 200m$. The whole spectrum is divided into 5 channels. Each node needs to find an available one to access. The transmission range and the number of nodes are dynamic parameters. We apply the random way point model, with different values of maximum speed. For each set of the maximum speed, we pick random pause times from the range. For simplicity, we set the average values for TS, TD, TC and TM, which are listed in Table 1. The maximum number of spectrum maps that a node can store randomly varies from 2 to 6.

5.1.2 Spectrum map model

The network is divided into grids, and the size of each grid is $10m \times 10m$. There are 50 PUs randomly distributed in the network. We assume that the PU active ranges, or spectrum availabilities in one grid, are the same. Therefore, each grid is associated with a spectrum map. We construct a virtual database which contains the spectrum maps at all grids. Nodes in one grid need to get the corresponding spectrum map. Otherwise, the spectrum sensing is applied. If a node moves to another grid, then the new spectrum map is needed. The probabilities of successful connections among nodes and the spectrum database are varied from 0.8 to 1.0. If more than one nodes connect to the database at

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2832021, IEEE Transactions on Network Science and Engineering

TABLE 2
Number of clusters

Number of nodes	20	25	30	35
Average cluster size	2.7	2.96	3.4	3.67
Number of nodes	40	45	50	55
Average cluster size	3.81	3.85	4.2	4.39

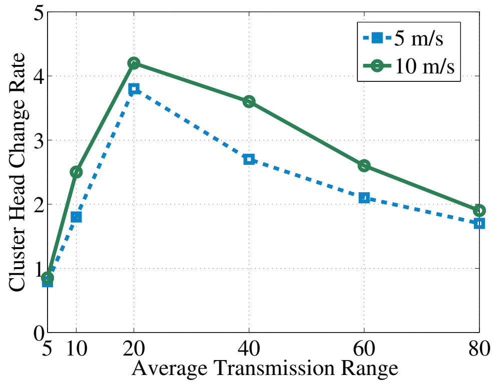


Fig. 5. Cluster head change rate.

the same time and they are within each other's transmission range, they would adopt the IEEE 802.11 backoff algorithm with the unit window length as TD.

5.1.3 Evaluation metrics

We first evaluate the reliability of our proposed hierarchical structure by changing the number of nodes and average transmission range of each node. We show the differences by varying the two parameters: the number of cluster sizes and the rate of cluster head changes.

Then, we evaluate the spectrum access performances under the hierarchical structure from the following two aspects:

- Access Scheme distributions: the distributions among the hybrid spectrum access scheme, which are sensing-based and spectrum map-based;
- Average time cost: the average time cost for spectrum access for each node in a cluster. We compare our results with two non-hybrid schemes.

Finally, we study the influences of the parameter K in Algorithm 1. We vary the value of K in the range of $[2, 6]$.

5.2 Simulation Results

The simulation results are presented based on the above three metrics.

5.2.1 Hierarchical structure influences

We show the average cluster sizes by changing the number of nodes in Table 2. When the number of nodes increases, the average cluster size also increases because of the larger node density. To show the rate of cluster head changes, we vary the average transmission range of each node from $5m$ to $80m$, and count the average number of cluster head changes per second. We compare two different maximum moving speeds $5m/s$ and $10m/s$. The results are shown in Fig. 5. When the average transmission range increases, the number of cluster head changes first increases, and then decreases. This is because when the transmission range is very small, the number of nodes in each cluster is relatively low. Therefore, there are fewer changes of cluster heads. When the transmission range increases, the cluster head rate increases. However, the transmission range becomes larger, each node needs to move a longer distance to cause the cluster head change. Moreover, through comparing the two lines in Fig. 5, the cluster heads change faster when the maximum moving speed is larger.

5.2.2 Spectrum access performances

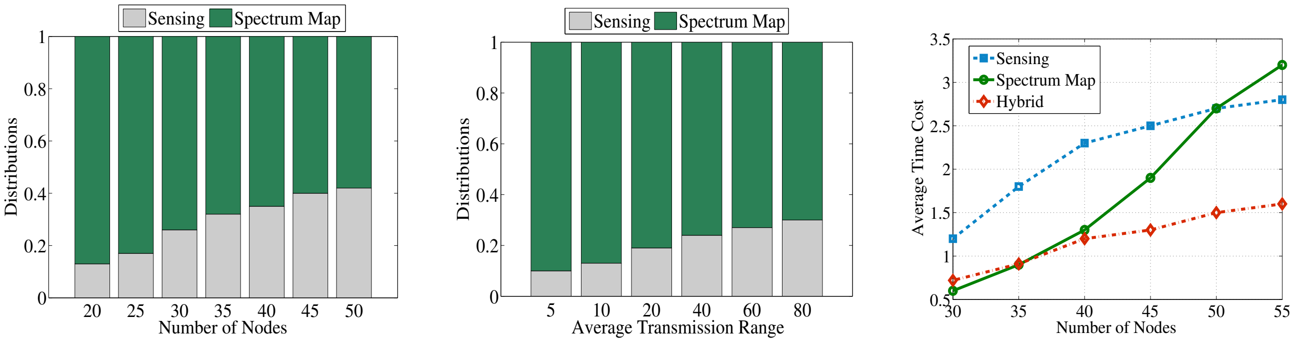
We first evaluate the distributions of two spectrum access schemes: sensing-based and spectrum map-based. We change the number of nodes and show the distribution differences in Fig. 6(a). The influences of varying transmission ranges to the access scheme distributions are shown in Fig. 6(b). When either the number of nodes or the average transmission range increases, the proportion of nodes adopting the spectrum map based scheme becomes smaller. This is because in both situations, the average cluster size becomes larger and each cluster head needs to serve more nodes. The number of nodes that are unable to fetch the corresponding information from cluster heads increases. Moreover, in both figures, the proportion of nodes adopting the spectrum map-based scheme reduces more slowly when the number of nodes or the average transmission range becomes larger.

In Fig. 6(c), we compare the average time cost of spectrum access among three schemes: our hybrid scheme under the hierarchical structure, the sensing-based scheme, and the spectrum map-based scheme. From the results, we can see that when the number of nodes increases, the average time cost increases for all three schemes. At the beginning, the sensing based scheme has the highest time cost among the three. This is because nodes may perform spectrum sensing multiple times since it has no information about the spectrum availabilities. When the nodes increase in number, the spectrum map-based scheme takes a longer time to access the spectrum. This is because more mobile nodes send requests to the database, and each session to reach the database faces more potential conflicts, which results in a longer time cost. Our hybrid scheme with the hierarchical structure is affected less when the number of nodes increases, since only cluster heads connect to the database and there are fewer conflicts.

5.2.3 Influences of different algorithm parameters

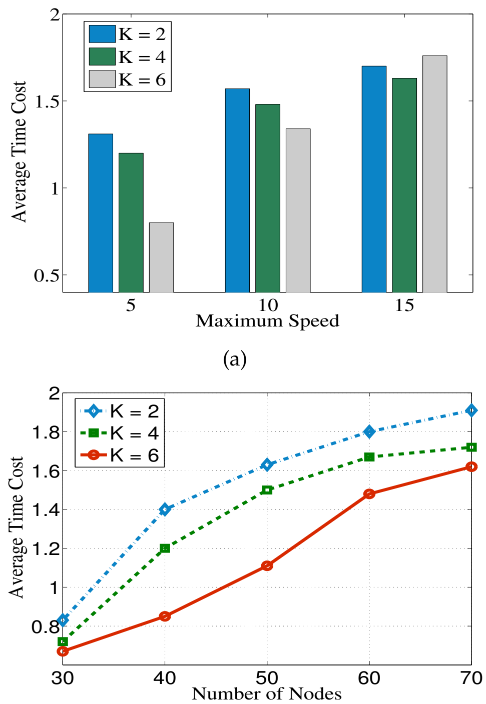
We show the results of three K values in Fig. 7. In Fig. 7(a), we change the maximum speed from $5m/s$ to $15m/s$. The average time cost increases under all three K values. At the beginning, the larger K value shows lower time cost. However, when nodes move with the maximum speed $15m/s$, $K = 6$ results in a longer time cost compared to $K = 4$. This is because when the maximum speed increases, the hierarchical structure evolves faster. In Fig. 7(b), $K = 6$ shows the best performance on the average time cost among the three settings. However, the difference of the average time cost reduces between $K = 4$ and $K = 6$ when the number of nodes increases. This is because cluster heads change more frequently.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2832021, IEEE Transactions on Network Science and Engineering



(a) (b) (c)

Fig. 6. Comparison of spectrum access performances under different settings.



(a)

(b)

Fig. 7. Influences of different K values on average time cost.

6 Conclusion

We proposed a mobility-compatible spectrum access framework in a database-assisted environment. We considered the situation where not every node was able to reach the database and proposed a hybrid spectrum access scheme based on our hierarchical structure. We selected cluster heads through their relative mobility metrics, storage limits, and connection qualities to the database. Only cluster heads connected to the database and provided assistances to nodes in their clusters. We formulated the fetching process on each cluster head as an MDP. In addition, the mobility influences were taken into account. We studied the evolutions of our hierarchical structure and presented a practical algorithm for each cluster head's decision on the best spectrum maps to download. Furthermore, we conducted simulations for evaluating our framework. As future works, we are studying the performance on large networks with more practical communication overhead.

Acknowledgments

This research was supported in part by NSF grants ECCS 1231461, ECCS 1128209, CNS 1138963, CNS 1065444, CNS-1564128, and CCF 1028167.

References

[1] G. Nair, Y. Moorthy, and S. Pillai, "A survey on dynamic spectrum sharing using game theory in cognitive radio networks," *International Journal of Research and Engineering*, 2016.
[2] Q. Zhao and S. B.M., "A survey of dynamic spectrum access," *IEEE Signal Processing Magazine*, 2007.
[3] G. Gur, S. Bayhan, and F. Alagoz, "Cognitive femtocell networks: an overlay architecture for localized dynamic spectrum access [dynamic spectrum management]," *IEEE Wireless Communications*, 2010.
[4] S. H. R. Bukhari, M. H. Rehmani, and S. Siraj, "A survey of channel bonding for wireless networks and guidelines of channel bonding for futuristic cognitive radio sensor networks," *IEEE Communications Surveys Tutorials*, 2016.
[5] A. A. Khan, M. H. Rehmani, and M. Reisslein, "Cognitive radio for smart grids: Survey of architectures, spectrum sensing mechanisms, and networking protocols," *IEEE Communications Surveys Tutorials*, 2016.
[6] A. E., L. G., L. E.G., and P. H.V., "Spectrum sensing for cognitive radio : State-of-the-art and recent advances," *Signal Processing Magazine, IEEE*, 2012.
[7] A. Ghasemi and E. S. Sousa, "Spectrum sensing in cognitive radio networks: requirements, challenges and design trade-offs," *IEEE Communications Magazine*, 2008.
[8] F. AB., G. R.E., K. E.W., P. Ecclesine, and P. S., "IEEE 802.11af: A standard for TV white space spectrum sharing," *IEEE Communications Magazine*, 2013.
[9] D. Ying and W. Jie, "Integration of spectrum database and sensing results for hybrid spectrum access systems," in *Proc. of IEEE MASS*, 2015.
[10] M. Souryal, M. Ranganathan, J. Mink, and N. E. Ouni, "Real-time centralized spectrum monitoring: Feasibility, architecture, and latency," in *Proc. of IEEE DySPAN*, 2015.
[11] Y. Xu, J. Wang, Y. Xu, Q. Wu, and A. Anpalagan, "Database-assisted spectrum access in dynamic networks: A distributed learning solution," in *CoRR*, 2015.

[12] M. R., C. R., M. T., and B. P., "SenseLess: A database-driven white spaces network," *IEEE Transactions on Mobile Computing*, 2012.

[13] C. Xu and J. Huang, "Game theoretic analysis of distributed spectrum sharing with database," in *Proc. of IEEE ICDCS*, 2012.

[14] H. B. Yilmaz, T. Tugcu, F. Alagz, and S. Bayhan, "Radio environmentmap as enabler for practical cognitive radio networks," *IEEE Communications Magazine*, 2013.

[15] O. J., Poikonen, J., and R. Wichman, "Effect of geolocation database update algorithms to the use of TV white spaces," in *Proc. of Crowncom*, 2012.

[16] A. Andreas, R. Janne, and M. Petri, "Improving accuracy for TVWS geolocation databases: Results from measurement-driven estimation approaches," in *Proc. of IEEE DySPAN*, 2014.

[17] Z. Kexiong, R. S. Kondaji, and Y. Yaling, "Location robustness in database-driven white spaces network," in *Proc. of IEEE DySPAN*, 2014.

[18] Z. Gao, H. Zhu, Y. Liu, M. Li, and Z. Cao, "Location privacy in database-driven cognitive radio networks: Attacks and countermeasures," in *Proc. of IEEE INFOCOM*, 2013.

[19] B. Gao, J.-M. Park, and Y. Yang, "Supporting mobile users in database-driven opportunistic spectrum access," in *Proc. of ACM MobiHoc*, 2014.

[20] F. Georgios, K. Michalis, P. Maria, S. Nick, and P. Maria, "Supporting wireless access markets with a user-centric qoE-based geo-database," in *Proc. of ACM MobiArch*, 2012.

[21] Y. Dai, J. Wu, and C. Xin, "Virtual backbone construction for cognitive radio networks without common control channel," in *Proc. of IEEE INFOCOM*, 2013.

[22] S. Liu, L. L., and K. M., "Cluster-based control channel allocation in opportunistic cognitive radio networks," *IEEE Transactions on Mobile Computing*, 2012.

[23] O. M. and A. O.B., "Event-driven spectrum-aware clustering in cognitive radio sensor networks," in *Proc. of IEEE INFOCOM*, 2013.

[24] L. Wu, X. Du, J. Wu, and B. Song, "Achieving fair spectrum allocation for co-existing heterogeneous secondary user networks," in *Proc. of IEEE ICCCN*, 2017.

[25] K. Chi, L. Wu, X. Du, G. Yin, J. Wu, B. Ji, and X. Hei, "Enabling fair spectrum sharing between wi-fi and lte-unlicensed," in *Proc. of IEEE ICC*, 2018.

[26] L. Huang, G. Zhu, and X. Du, "Cognitive femtocell networks: an opportunistic spectrum access for future indoor wireless coverage," *IEEE wireless communications*, 2013.

[27] B. P., K. N., and L. T. D. C, "A mobility based metric for clustering in mobile ad hoc networks," in *International Conference on Distributed Computing Systems Workshop*, 2001.

[28] J. Wu and F. Dai, "Virtual backbone construction in MANETs using adjustable transmission ranges," *IEEE Transactions on Mobile Computing*, 2006.

[29] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing*, 2002.

Ying Dai received her B.Eng. in Software Engineering from Nanjing University, Nanjing, China, in 2010. She is currently a Ph.D. candidate in the Department of Computer and Information Sciences, Temple University, USA. Her research interests include various topics in cognitive radio networks. She is now working on spectrum sensing, efficient channel assignment, and data transmission in cognitive radio networks. Her publications include journal papers in IEEE TPDS, TVT, and conferences papers in IEEE Infocom, MASS, and ICCCN.

Jie Wu is the chair and a Laura H. Carnell professor in the Department of Computer and Information Sciences at Temple University. He is also an Intellectual Ventures endowed visiting chair professor at the National Laboratory for Information Science and Technology, Tsinghua University. Prior to joining Temple University, he was a program director at the National Science Foundation and was a Distinguished Professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. Dr. Wu regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Service Computing and the Journal of Parallel and Distributed Computing. Dr. Wu was general co-chair/chair for IEEE MASS 2006, IEEE IPDPS 2008, IEEE ICDCS 2013, and ACM MobiHoc 2014, as well as program co-chair for IEEE INFOCOM 2011 and CCF CNCC 2013. Currently, he is serving as general chair for ACM MobiHoc 2014. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a CCF Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.

Dr. Xiaojiang (James) Du is currently a professor in the Department of Computer and Information Sciences at Temple University. Dr. Du received his B.S. and M.S. degree in electrical engineering from Tsinghua University, Beijing, China in 1996 and 1998, respectively. He received his M.S. and Ph.D. degree in electrical engineering from the University of Maryland College Park in 2002 and 2003, respectively. Dr. Du was an Assistant Professor in the Department of Computer Science at North Dakota State University between August 2004 and July 2009, where he received the Excellence in Research Award in May 2009. His research interests are security, cloud computing, wireless networks, computer networks and systems. He has published over 270 journal and conference papers in these areas. Dr. Du has been awarded more than $\$3M$ research grants from the US National Science Foundation (NSF), Army Research Office, Air Force Research Lab, NASA, the Commonwealth of Pennsylvania, and Amazon. He serves on the editorial boards of four international journals. Dr. Du will serve as the Lead Chair of the Communication and Information Security Symposium of the IEEE ICC 2015, and a Co-Chair of the Mobile and Wireless Networks Track of the IEEE WCNC 2015. He was the Chair of the Computer and Network Security Symposium of the IEEE/ACM International Wireless Communication and Mobile Computing conference 2006 - 2010. He is (was) a Technical Program Committee (TPC) member of several premier ACM/IEEE conferences such as INFOCOM (2007 - 2019), IM, NOMS, ICC, GLOBECOM, WCNC, BroadNet, and IPCCC. Dr. Du is a Senior Member of IEEE and a Life Member of ACM.