Distributed Localized Interference Avoidance for Dynamic Frequency Hopping ad hoc Networks

Sebastian Koslowski, Jens P. Elsner, Friedrich K. Jondral; Karlsruhe Institute of Technology (KIT)
{koslowski, elsner, jondral}@kit.edu

Stefan Couturier; Fraunhofer Institute for Communication, Information Processing and Ergonomics
stefan.couturier@fkie.fraunhofer.de

Cédric Keip, Olaf Bettinger; Thales Defence & Security Systems GmbH
{cedric.keip, olaf.bettinger}@thalesgroup.com

***Abstract*—We present a cognitive spectrum access solution for FH-CDMA ad hoc networks. Building on a cluster-based split phase multi-channel MAC protocol, we propose a mechanism for local interference avoidance through distributed hopset adaptation. Its goal is to identify and substitute channels not suitable for reliable communication. Substitution rules replace channels by locally unused hopsets. This way, interference is mitigated while maintaining orthogonality between nodes' hopsets. We evaluate the gains of the proposed method with a simplified system model.**

I. Introduction

Wireless ad hoc networks, as they might find application in military communications, require a multi-channel medium access architecture and high robustness against external interference. A multi-channel capability allows to support a high number of network nodes, while frequency hopping is a proven way to increase link robustness. A combination of both techniques is studied in the following for a clusterized ad hoc network. The main novelty is a proposal to avoid bad channels by substituting them with locally orthogonal hopsets, thereby avoiding external interference while minimizing network internal (self-)interference.

The goal of the *CORASMA* (Cognitive Radio for Dynamic Spectrum Management) study is to create a versatile simulation framework for the study, analysis and evaluation of cognitive radio solutions for military ad hoc networks under various environmental conditions. Different solutions are built upon a common system called *basic waveform* and extend its features by a different cognitive process. The CORASMA cognitive solution presented in this paper adds adaptive frequency hopping to the basic waveform. The focus of our work is not on the basic waveform itself but rather on the cognitive solution build on top of it. Therefore we confine ourselves to outlining the functionality of the basic waveform relevant for our cognitive solution.

Adaptive frequency hopping is a well-studied approach [1]–[3] for the mitigation of fixed frequency interference. It has been included in different communication standards, e.g. Bluetooth [4]. Although the network structures of the CORASMA system and Bluetooth differ in several aspects, the basic idea of excluding bad channels from the hopset is the same. Both schemes organize nodes into clusters and use a special node per cluster for the planning and coordination of channel access and hopset adaption. However, the different piconets that makeup a Bluetooth scatternet are neither guaranteed to use orthogonal hopsets or to be synchronized. In contrast, our approach uses a multi-channel MAC in which the individual clusters do not interfere - provided enough resources are available.

The rest of this paper is organized as follows: Section II outlines the clusterized system architecture of the CORASMA basic waveform, while Section III states your interference avoidance approach. Section IV offers an analysis of the adaptation overhead/throughput trade-off. Conclusions are presented in Section V.

II. System Architecture

A. CORASMA ad hoc network

A cognitive waveform in CORASMA consists of the components inherited by the basic waveform and those which make up the *cognitive manager*. These include, for example, components to perform spectrum sensing tasks, to adjust transmission parameters or to modify the behavior of the link or network layer.

The basic waveform is capable of setting up and running a wireless ad hoc network. This includes building and maintaining neighborhood information, organizing access to the available channels for data transmission, as well as routing and forwarding of messages with different priorities. On the physical layer a SC-FDMA / OFDMA scheme is employed for the communication between nodes. For medium access control a multi-channel protocol is used: The network operates on a set of physical channels allowing multiple nodes to communicate at the same time (FDMA). A split-phase approach [5] is used to create a dedicated channel for the exchange of control information. The dedicated channel is common to all nodes in the network and uses a slotted random access scheme corresponding to a series of time slots called *Random Access Slots* (RAS). It is assumed that all nodes are equipped with only a single transceiver and therefore data transmissions cease during the RAS. The nodes switch to the channel of RAS and exchange control messages necessary for building and maintaining the network and its structure.

B. Cluster structure

Based on a neighbor discovery mechanism executed over the RAS channel, all nodes are organized in a clustered structure.

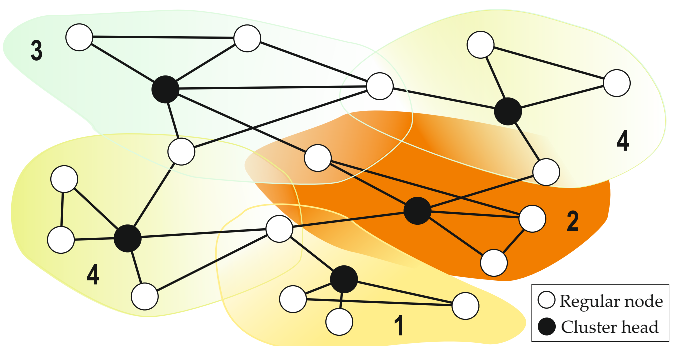

Fig. 1. Exemplary network structure with four clusters

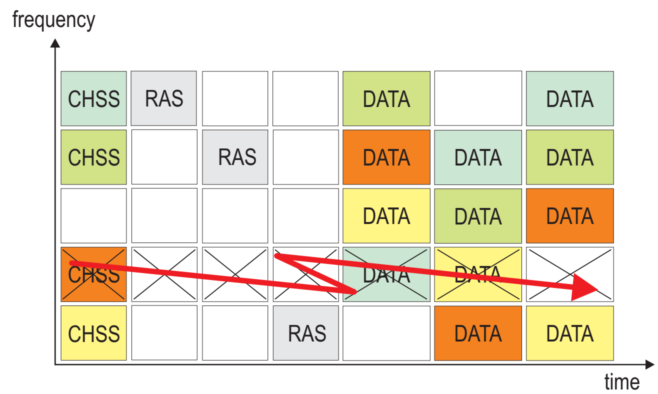

Fig. 2. Frequency hopping scheme with one interfered channel

Each cluster operates on a dedicated logical channel and one node is elected as *cluster head* (CH). It manages access to this cluster channel for all its members by receiving and granting resource reservation requests during the RAS. The initial selection and spatial reuse of cluster channel frequencies is driven by a distributed graph coloring algorithm. Cluster heads are elected with the objective to have at least one within reach of every node. As shown in Figure 1, no two CHs are allowed to be neighbors. Hence, CHs are always 2-hop neighbors. To allow for inter-cluster communication some nodes function as gateways. These belong to two or more clusters and switch back and forth between the channels of the respective clusters.

Within each cluster, the communication comprises a TDM system for control and data transmissions as shown in Figure 2. A special slot called Cluster Head Signaling Slot (CHSS) at the beginning of each MAC frame is used by the CH to announce the resource allocation within the next data period and convey signaling information to all cluster members. The CHSS is followed by a certain number of RAS, which allows nodes to send resource reservation requests to their respective cluster heads. Next is the data phase, in which the cluster channels are divided into several time slots. Each of them comprises a number of subbands allowing the nodes to exchange information according to the resource allocation conveyed via the CHSS.

C. Orthogonal hopsets

As part of our work we introduce a FH-CDMA system to the basic waveform. The logical channels of the MAC are no longer mapped directly onto physical channels $\Gamma = \{c_0, c_1, ..., c_{N-1}\}$. Instead, each of them is assigned one of K hopsets $(h_{k,n})_{n=-\infty}^{\infty}$ with $h_{k,n} \in \Gamma$, $0 \leq k < K$ and the time index n, which increases with each slot. To prevent additional internal interference, these hopsets are constructed to be orthogonal:

$$\forall k, \forall n \neq m : h_{k,n} \neq h_{k,m}, \tag{1}$$

hence, no two clusters occupy the same physical channel at the same time. The non-localized logical channels, e.g. the random access slots, are assigned a separate hopset which must be globally known and unique to allow for cluster independent communication. This enables nodes to join the network, re-join after connection loss and also the merging of networks.

To obtain orthogonal hopsets in a distributed manner, a sequence of permutations $(\Pi_n^\Gamma)_{n=-\infty}^{\infty}$ of the available channels is constructed at each node. Orthogonality is only guaranteed if all nodes choose the same permutations and select a different entry:

$$\Pi_n^\Gamma = (h_{0,n}, \cdots, h_{K-1,n}) \in \Gamma^K \tag{2}$$

The Knuth Shuffle algorithm [6] allows the calculation of permutations based on a set of random numbers: For each permutation K random values $r_k \in \mathbb{N}_0$ with $r_k \leq k$ are used to shuffle the list of channels:

$$\tilde{\Gamma}_k^{(0)} = c_k \tag{3}$$

$$\tilde{\Gamma}_k^{(i+1)} = \begin{cases} \tilde{\Gamma}_{r_i}^{(i)} & \text{, for } k = i+1 \\ \tilde{\Gamma}_{i+1}^{(i)} & \text{, for } k = r_i \\ \tilde{\Gamma}_k^{(i)} & \text{, else} \end{cases} \tag{4}$$

$$\Pi_n^\Gamma = \left(\tilde{\Gamma}_0^{(N-1)}, \cdots, \tilde{\Gamma}_{N-1}^{(N-1)}\right) \tag{5}$$

To derive the same sequence of permutations Π_n^Γ on different nodes, the values for r_k must be taken from synchronized pseudo-random number generators. This can be achieved by setting a common initial seed at defined time on each node; e.g. derived from a known hopping key.

Although frequency hopping increases the PHY and MAC layer complexity, it offers many advantages with regard to mitigation of both internal and external interference. As the clusters' hopsets are spread over all available frequencies, the penalty for a violation of the assumed perfect local orthogonality is reduced. Especially if cluster nodes are mobile, thus creating constant changes in the interference environment, the use of frequency hopping mitigates the chance of overlapping time-frequency slots. The influence of external interference, which renders certain frequencies unusable for communication is also reduced as shown in Figure 3. Depending on the number of channels available, frequency hopping provides additional transmission security by obfuscating individual data links.

III. Interference avoidance through hopset adaptation

A. Basic mechanism

To further improve the robustness introduced by frequency hopping and to allow vacating physical channels occupied by primary users, the hopsets should be adaptable to respond to changes in the interference environment. A global adaptation of certain hopsets is unfeasible especially for localized interference. In addition, the amount of signaling overhead and the required delay renders network global hopset adaptation impractical.

Instead, each cluster shall decide for itself when and how to alter its hopset and how to make use of the locally available channels. To avoid additional internal interference with neighboring clusters, hopset changes must be performed in a cooperative manner. This requires each cluster to exchange information about the currently used hopsets with all its neighbors and enables the respective cluster heads to identify feasible changes to their hopsets.

A simple way of altering a hopset $(h_{k,n})$ would be to substitute a bad channel $c_{\text{bad}} \in \Gamma_{\text{bad}} \subset \Gamma$ with another channel $c_{\text{alt}} \notin \Gamma_{\text{bad}}$. But, as the neighboring clusters continue hopping over all available channels Γ this strategy leads to additional internal interference:

$$\exists n, l : h_{k,n} = c_{\text{bad}} \wedge h_{l,n} = c_{\text{alt}} \tag{6}$$

Instead, a sequence of channels must be found that are guaranteed not to collide with neighboring clusters. As all hopsets are orthogonal, any locally unused hopset is such a sequence of channels and can be used to substitute a bad channel. We define the substitution function S, which maps each channel $c_k \in \Gamma_{\text{bad}}$ onto the hopset index k of an unused hopset $(h_{k,n})$:

$$S : \Gamma_{\text{bad}} \to [0, K) \tag{7}$$

When determining a node's next hop, the current channel permutation Π_n^{Γ} defined in (2) is transformed as follows:

$$\tilde{h}_{k,n} = \begin{cases} h_{k,n} & \text{, for } h_{k,n} \notin \Gamma_{\text{bad}} \\ h_{S(h_{k,n}),n} & \text{, else} \end{cases} \tag{8}$$

For multiple bad channels, (8) has to be applied recursively as a substitution hopset might point to another bad channel $h_{S(h_{k,n}),n} \in \Gamma_{\text{bad}}$.

To prevent possible collisions, neighboring clusters must not use the same hopset for their substitutions at the same time. This requires each cluster head to not only know the hopsets used by its neighboring clusters, but to also be informed of any substitutions they perform. Figure 3 shows an exemplary setup with $|\Gamma| = 11$ hopsets and their usage by the clusters depicted in Figure 1. For simplicity, the hopset and time slots for the network's control channel are not included. As an example, imagine that the blue cluster, which uses hopset $(h_{3,n})$, recognizes that channel c_5 is bad. It decides to substitute c_5 with hopset $(h_{7,n})$. This means that at $n = 6$ the blue cluster hops on c_2 instead of c_5, and at $n = 10$ it hops on c_8. All other times it follows $(h_{3,n})$.

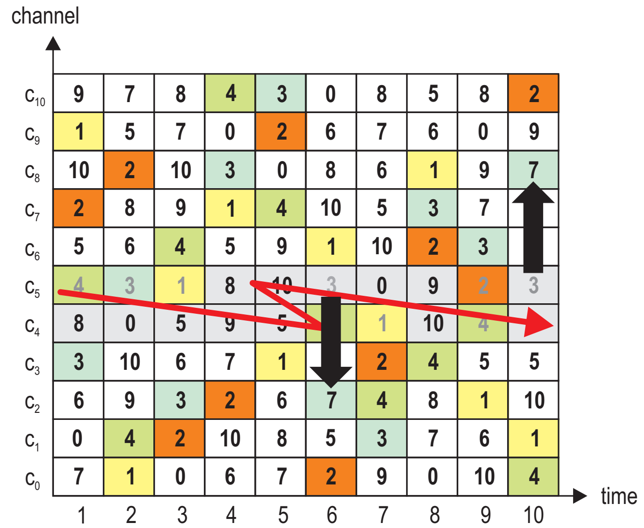

Fig. 3. Substitution of bad channels (c_4, c_5) in FH-CDMA scheme

A substitution hopset can be interpreted as a resource accessible by different clusters in a time division multiplexing scheme. Because there is always no more than one cluster affected by a certain bad channel at the same time, neighboring clusters can use the same substitution rule if they also find this channel to be bad. This way the substitution hopset can be shared among different clusters without collisions. For example, in Figure 3 the cluster on hopset $(h_{1,n})$ may also choose to substitute c_5 with hopset $(h_{7,n})$ without interfering with the cluster on hopset $(h_{3,n})$.

B. Sensing

In addition to resource management, the cluster heads are in charge of conducting interference measurements and collecting them from their peers. On each node sensing is divided into implicit and explicit sensing.

1) Implicit sensing: Implicit sensing uses physical channel state information and the packet error rate to monitor all of the currently used channels. If the packet error rates of all links are significantly higher on a certain channel than on the others, this frequency is assumed to be experiencing some form of interference – external or internal.

2) Explicit sensing: As implicit sensing only observes the non-substituted channels, it can only be used to exclude channels. Therefore, explicit sensing is needed to examine channels, which have been previously removed from the hopset.

The cluster head must organize this form of sensing by allocating sensing slots in the data phase. During these slots all nodes assess the state of excluded channels. Sensing slots are allocated periodically, however, if a bad channel continues to be unsuitable for re-inclusion it is checked less often. An exponential back-off can be used to determine the sensing update rate for each excluded channel.

3) Collaborative Sensing: Both implicit and explicit sensing results are combined on each node resulting in a reliability value for each channel. Upon request these sensing results are sent to the cluster head which collects and merges them. Based on the overall results the cluster head may choose to exclude or re-include a channel by finding a suitable substitution rule or withdrawing one, respectively.

C. Hopset adaptation

Scheduling and controlling the different processes, which make up the observing of and reacting to changes in the radio environment, characterize the behavior of the cognitive manager. On the one hand a frequent allocation of sensing slots and an agile policy on adapting hopsets promise best use of the available spectral resources. On the other hand the overhead necessary to accomplish these tasks reduces the capacity for data transmissions.

The planning, negotiating and executing of hopset adaptations is triggered by updates to sensing results. However, not every update to this list has to result in a new adaptation request. Changes to a cluster's hopset must be limited to prevent the associated protocol messages from causing too much overhead. The hopset adaptation protocol itself takes at least two MAC frames to execute. Additional time has to be provided for scheduling delays, queuing and retransmissions. This has to be taken into account when limiting the number of concurrent adaptations.

To organize and handle hopset adaptations a two step protocol is used. When a cluster decides to alter its hopset, a solution is built based on the local knowledge on the neighboring clusters' hopsets. A solution contains one or more channels from Γ_{bad}, which will be substituted by alternative, locally available hopsets. This information is spread as a broadcast to alert all nodes of the impending adaptation. Gateway nodes relay this *HopsetAdaptationRequest* to their other cluster heads. Each request contains a randomly chosen request id and the current time to limit its validity.

If a conflicting, older request arrives at a cluster head, it will drop its own. If two requests have been issued at the same time, the request id is used to drop one of them. If a request is still valid after a certain time, a *HopsetAdaptationConfirmation* is sent to all nodes in the cluster and the aforementioned neighboring cluster heads. This message references the previously sent *HopsetAdaptationRequest* and contains the exact time when the hopset adaptation will be applied.

If a node has missed a *HopsetAdaptationRequest*, for example due to lost packets or transmission time-outs, the node might miss a hopset substitution. It continues on the stale hopset and will miss a hop, but will still be able to communicate within the cluster. To assure re-synchronization, the current hopset substitution list should be broadcast periodically in the cluster head signaling slot.

IV. Analysis

In this section the gain of the proposed hopset adaption method is evaluated by comparing its performance with a fixed, random hopping system. For the simulation a simplified model is derived. We consider N fixed-positioned, neighboring clusters in a system with K channels and make the following assumptions:

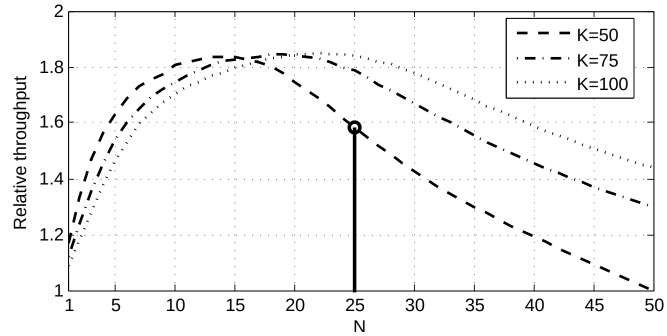

Fig. 4. Performance gain of hopset adaption vs. random hopping for varying number of clusters N ($p_s = p_e = 0.05$, $N_e = 5$)

- For each channel an independent two-state discrete Markov model is used to set its usability. An interference starts and ends with a probability p_s and p_e, respectively. The interference duration hence follows a geometric distribution and its average is given by $1/p_e$. The expected number of unusable channels is $Kp_s/(p_e + p_s)$. All clusters in the neighborhood experience the same interference.
- The detection probability for implicit and explicit sensing is $p_d = 0.95$. An interference can be detected within a single hop on the respective channel. Explicit sensing is carried out periodically. Each time step N_e channels are examined by the nodes of the clusters.
- Ideal selection of substitution hopsets. Whenever an interference is detected by any cluster in the vicinity the resulting substitution will be adopted by all other clusters, i.e. an interference on a certain channel is substituted by the same hopset in every cluster.

Using these assumptions the behavior of the system can be simulated by tracking the state of each channel as well as the substitutions made by the clusters. For the following results 10^5 iterations are used.

Figure 4 shows the performance gain of our proposed scheme over fixed, random orthogonal hopping for an exemplary interference situation. The relative throughput is calculated by counting the number of usable hops with adaptive hopping and dividing it by the number of usable hops with random orthogonal hopping. Using $p_s = p_e$, on average $K/2$ channels are bad. Hence, $N \geq K/2$ results in a congestion of the system. In this case, no more unused hopsets can be found and therefore hopping on bad channels is unavoidable. For large N the performance converges to that of random orthogonal hopping. For each K there is a number of clusters N for which the relative average gain is maximal. The maximum lies to the left of $K/2$, the mean number of bad channels. If there are only a few clusters, the probability of using bad channels reduces together with the need and potential gain from substituting them.

Figure 5 shows the average performance gain for different interference statistics. The marked position corresponds to that marked in Figure 4. It can be seen that the substitutions are most effective for a long mean interference duration. However,

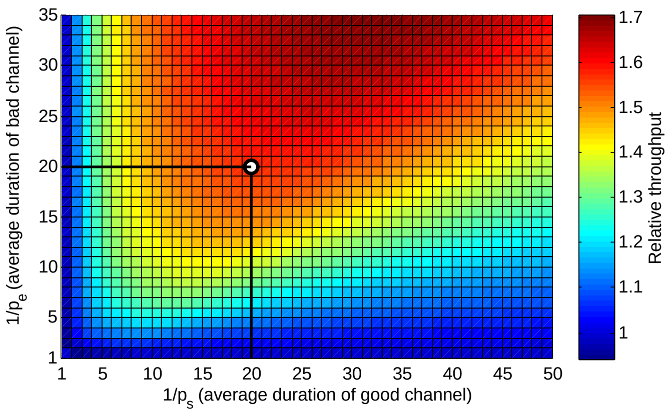

Fig. 5. Performance gain of hopset adaption vs. random hopping for different interference statistics ($N = 25$, $K = 50$, $N_e = 5$)

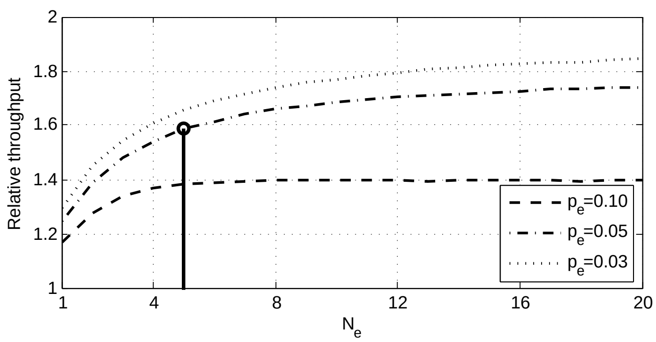

Fig. 6. Performance gain of hopset adaption vs. random hopping for varying intensity of explicit sensing N_e ($N = 25$, $K = 50$, $p_s = 0.05$)

when interference becomes more seldom, the efficiency is reduced. For channels with frequent but short interference the performance gain is also limited. Especially in real systems, additional limitations are caused by the signaling and sensing overhead necessary for executing the substitutions.

Figure 6 shows the average performance gain for different N_e. The average gain increases with the number of channels N_e that can be explicitly sensed simultaneously. As can be seen in the figure, depending on p_e, a small number of explicitly sensed channels suffices to achieve maximum gain. These findings suggest that the overhead associated with sensing in a practical system need not be very large.

Note that the same point on the simulation curves has been marked with a circle in all figures.

V. Conclusion

We studied the complexities of introducing frequency hopping in a synchronized clusterized ad hoc network and provided a solution for a distributed adaptation of frequency hopping sequences. The approach offers robustness through local interference avoidance and at the same time keeps the orthogonality to minimize self-interference. Our simulation results outlined the potential gains in the average throughput. Non-idealities of RF hardware such as oscillator settling times have so far been neglected in the study and will be subject of future work. The goal will be to derive reasonable hopping rates and channel bandwidths, which can be serviced by current SDR hardware. Furthermore, the simulation model will be extended to incorporate independent, time-varying channel statistics and a more realistic cluster behavior and performance evaluation.

Acknowledgment

This research work was carried out as part of *CORASMA: Cognitive Radio for Dynamic Spectrum Management – European Defence Agency (EDA) Project B-0781-IAP4-GC.*

References

[1] J. P. Elsner, R. Tanbourgi, J. Schmid, and F. K. Jondral, "Hopping strategies for adaptive fh-cdma ad hoc networks under external interference," in *IEEE International Conference on Communication*, 2012.

[2] P. Popovski, H. Yomo, and R. Prasad, "Strategies for adaptive frequency hopping in the unlicensed bands," *Wireless Communications, IEEE*, vol. 13, no. 6, pp. 60 –67, dec. 2006.

[3] ——, "Dynamic adaptive frequency hopping for mutually interfering wireless personal area networks," *Mobile Computing, IEEE Transactions on*, vol. 5, no. 8, pp. 991 –1003, aug. 2006.

[4] Bluetooth SIG, *Specification of the Bluetooth System Version 4.0 [Vol 0]*, http://www.bluetooth.com, 2010.

[5] J. Mo, H.-S. So, and J. Walrand, "Comparison of multichannel mac protocols," *Mobile Computing, IEEE Transactions on*, vol. 7, no. 1, pp. 50 –65, Jan. 2008.

[6] D. Knuth, *The Art Of Computer Programming, Volume 2: Seminumerical Algorithms, 3/E.* Pearson Education, 1981.