Hitchhiking in Cognitive Radio Networks: Spectrum Sensing Assisted By Cores and Clusters

Ying Dai and Jie Wu

Department of Computer and Information Sciences, Temple University
Motivation

- Spectrum sensing in Cognitive Radio Networks (CRNs): protect primary users
- Accuracy requirement + extra time cost
- Question: Can we reduce the sensing cost by having nodes help each other?

Total channel set: \{m_1, m_2, m_3\}
**Motivation**

- Hitchhiker: A node can make use of other nodes’ most recent sensing results to benefit its current sensing.
  - Two dimensions: location, time.
- Potential extra time cost during the information exchange among nodes may be harmful.
- Our solution: *Cores and Clusters!*
The phase before spectrum sensing happens: How to select channels for sensing in CRNs.

Our goal: Reduce the total number of channels that a node needs to sense before finding an available one.

Core structure: Each node designates a neighbor or itself as its core, and can gain help for the spectrum sensing phase.

Extension: A 2-layer structure of both clusters and cores, and the corresponding spectrum sensing scheme.
Core Construction

- Information exchange with neighbors.
  - Available channel set.
- Weight Calculation.
  - For a node $v$, the weight of it is: $W_v = \sum_u |M_u \cap M_v|$, $\forall u \in N_v$.
- Core designation.
Core Construction

An example:

\[ d_u = \{w, x\}, \quad d_v = \{u, y, s, z, v\}. \]
Spectrum Sensing With Cores

The help gained from the core is the channel list, which sorts the channel according to their available probability.

- On the node side: Pull, Sense, Transmit, Push.
- On the core side: Return, Update.
Core Evolution

• What if a node designates a wrong core?
  ○ The node and its core do not share the similar channel availabilities.

• The core evolution is necessary to fix this situation.

• But... How?
  ○ We need to find a way to evaluate the assistance provided by the core.
Core Evolution

- The basis for core updates:
- For a node $u$ and its core $v$ ($v = c_u$), $u$ needs to update $c_u$ if and only if $A_{uv} > A_u$.
  - $A_{uv}$ is the estimated average number of channels to sense if $u$ receives assistance from $v$.
  - $A_u$ is the estimated average number of channels to sense if $u$ senses itself $u$ and gains no assistance from others.
**Core Evolution**

- The evaluation is performed on the core side.
  - A node is unable to evaluate since it always senses based on the core’s information.
- Compare the performance with and without core’s information.
  - The core considers the virtual situation that if the node sends a request now, rather than pushing back its current channel information.
Core Evolution

Core reselection:

- After a node identifies its core needs to be reselected, it reselects a new core.
- Simply reselect from its remaining neighbors aside from the wrong core.
- Advantage of core structure: Easy to propagate!
Cluster-core Motivation

- Motivation: In a sparse network, the average help provided by each core is limited.
- To increase the performance, what about having more nodes in a longer distance share information?
- Cluster-core structure: Build clusters on top of the cores.
Cluster-core Construction

- Select cluster heads from current cores, using similar weight definition as for core selections.
  - $WC_v = \sum_u |M_u \cap M_v|, \forall u \in NC_v$.
  - $NC_v$: the set of $v$’s neighbor cores.
- An example of the mixed cluster-core structure is:
Cluster-core Construction

We apply the classical cluster head selection algorithm here:

1. All cores are initially uncovered;
2. An uncovered core becomes a cluster head, if it has the highest weight;
3. The selected cluster heads and their connected 1-hop neighbor cores are marked as covered;
4. Repeat Steps 2 and 3 on all uncovered cores.
Spectrum Sensing With Cluster-core

- Cluster head collects information from the cores and shares the information among multiple cores.

- The overview of the process:
  - Cluster heads: periodically collect from and send to the cores in the same cluster.
  - Cores: updates their corresponding channel information, and return the updated information to other nodes.
An example of spectrum sensing with cluster-core structure:

Cluster head: $v$, collects and shares the information. Cores: $u$, $v$, $u'$, update their channel list for other nodes.
Spectrum Sensing With Cluster-core

Some illustrations:

- Under the cluster-core structure, the work on the node side with its core remains unchanged.
- The cluster heads push to their cores, instead of having the cores pull from the cluster heads.
  - A cluster head usually has more members than a core.
- It is not true to claim that one of the core-only and cluster-core structures is always better than the other.
Simulation Settings

Main parameters:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of PUs</td>
<td>10</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>[50, 600]</td>
</tr>
<tr>
<td>Number of channels</td>
<td>[5, 20]</td>
</tr>
<tr>
<td>Single data task duration</td>
<td>3</td>
</tr>
<tr>
<td>Size constraints for cores</td>
<td>[1, 14]</td>
</tr>
<tr>
<td>Information exchange frequency for cores</td>
<td>[1, 3]</td>
</tr>
<tr>
<td>Information exchange frequency for clusters</td>
<td>[3, 9]</td>
</tr>
</tbody>
</table>

1. Parameters to vary: size constraints of the core-only structures, information exchange frequencies between a node and its core.

2. Performance to compare: the average number of channels to sense
Different size constraints for core-only structures VS Random sensing
Simulation Results

Cluster-core structure VS core-only structure VS Random sensing

[Graphs showing comparison]
Conclusions

- Our focus: how to select channel for spectrum sensing.
- Two structures: core-only and cluster-core structures.
- The evolution process for the core-only structure.
- Two corresponding sensing schemes.
Thank you!
If you have any question, please contact Ying Dai (tuc74224@temple.edu).