Group creation in a collaborative channel allocation scheme

Subtitle

Hans Jørgen Furre Nygårdshaug

Thesis submitted for the degree of
Master in Informatikk: programmering og nettverk
60 credits

Institutt for informatikk
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2017

Group creation in a collaborative channel allocation scheme

Subtitle

Hans Jørgen Furre Nygårdshaug

Group creation in a collaborative channel allocation scheme

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

0.1 Disposition

0.1.1 Introduction part

- Introduction to the wifi interference problem
- Taking a step back and looking at other attempts on solving the problem
- Begin presenting Torleiv and Magnus work, the idea, and maybe the p2p protocol.
- End of by showing that there is a problem with creating, limiting and updating groups.

0.1.2 Main thesis part

The problem of data replicaton

- The problem.
- Possible solutions? References.
- Complexity, out of scope for thesis. Assume problem is solved.

The algorithm itself

- Elaborating on the problem, introducing the first algorithm suggestion.
- Explain simulation data creation with stochastic uniform distribution.
- Show how the group creation algorith was created, design decisions (iterations etc).
- Results with visualizations through the visualization tool.
- Evaluate results and consider improvements. How will this work in the wild?
- Introduce SSB data, the data format and why it is relevant. How is the tool made.
- Same procedure with result visualization and result evaluation. Do we still need improvements?
- Introducing Wigle as data source. Show results on map?

0.1.3 Concluding part

- Have we created a meaningful algorithm that can be implemented in hardware?

0.2 Channel assignment

To deal with the problem of channel assignment we will think of an access point as a vertex in a graph. When an access point scans its radio, it can see the strength of all nearby wireless networks measured in dBm (decibel milliwatts). This decibel value will be the value of the edge between one access point to another. With such a graph expressing a wireless network topology, it boils down to a graph coloring problem, where the number of colors represents the number of non-overlapping channels. Exactly how an algorithm can be designed to optimally distribute channels within the interfering topology, is out of the scope of this thesis. However we can define some invariants that has to be true for such an algorithm to work:

1. All access points has to run the same algorithm
2. All access points must run the algorithm on the same wireless network topology graph
3. Because of the complexity of the problem the number of access points in a network graph can not be too big

As not all access points in the world is relevant for the algorithm, and it can only perform well with a certain number of nodes, the system has to have a way to subdivide the world in groups. Ideally a group is contained within a geographical location where a certain number of nodes is aggregated. If the number of nodes in the group exceeds what the channel assignment algorithm can handle, the group has to be split into one or more new groups.

0.3 The Group Algorithm

0.4 Simulation data

Primarily, before beginning to implement and test the group creation algorithm, the task is to create usable data to perform testing on. . Additionally, each node should have a neighbour list, containing the respective interference measured in $-dBi$ for each neighbour. In addition, the following parameters should be variable depending on each test scenario:

- Map size (width of x- and y-axis).
- Number of nodes
- Minimum distance between nodes (in meters)

- Minimum measured $-dBi$ for a neighbour to consider it interfering

The program that generates the data is written in python, and can export the topology data to JSON-format so it can be visualized or used by other applications.

The interference levels between access points is calculated by iterating through each node. For each node N we record its x and y position, and then start a second iteration through the nodes. For each node in the second iteration n we calculate the distance d in meters between N and n using Euclidean distance.