Multi LTE Sites

Ohad Shpindel

This document define the procedure of connecting number of LTE stations to one spanning tree in order to enable data transformation among them

Contents

[1.1 Schemas 3](#_Toc508795642)

[1.1.1 System architecture schema 3](#_Toc508795643)

[1.2 Elements definition 4](#_Toc508795644)

[1.2.1 Node Object 4](#_Toc508795645)

[1.2.2 Queue 5](#_Toc508795646)

[1.2.3 Plotter 6](#_Toc508795647)

[1.2.4 Test bench xml file 8](#_Toc508795648)

[1.2.5 Scan\_mat xml file 8](#_Toc508795649)

[1.2.6 XLS-Visual basic 8](#_Toc508795650)

[1.3 Algorithm and Simulation 9](#_Toc508795651)

[1.3.1 Algorithm basic assumptions 9](#_Toc508795652)

[1.3.2 System Behavior 10](#_Toc508795653)

[1.3.3 Simulation illustration example 12](#_Toc508795654)

[1.4 Plotter GUI interface 14](#_Toc508795655)

## Schemas

### System architecture schema

Node Object

Test bench .xml

Once

Push

Pop

Queue

Edit

XLS  
VB

Scan\_mat.xml

## Elements definition

### Node Object

Each node defined by python class with these parameters:

Coordination- (x,y)

CID- IMSI (3 last digit)  
  
mode- root/client

RF channel

state- N/A, seeking, connected, tree connected (n\_conn)

Tree\_No  
Tree\_rate- treeRate= (no. of nodes in the tree)\*1000+RNid  
\* "RNid" is the CID of the tree's root node  
user\_load- number of connected users (0-16)  
nextHopCID - IMSI of the node's next hop  
scan\_mat- matrix that contains the distance from each other node on the plot   
 (parallel to RSSI in a real system)  
startup\_time- random startup time in secs (5-10)

When node is in root mode (RN) all the system info tunneled to it and then policy decisions are made.

The policy decisions are forward to the Queue for plotting

Class functions:

scanArea()- scans the area for available node connections returns scan\_matrix

openRF()- open RF transmission

openSocket()- opens socket to receive connections

### Queue

The queue is a tool that sends objects to the plotter to be shown, the RN collects all the data from the other nodes and passes it to the plotter via the queue.

### Plotter

A process that plots the current state on the screen in real time.

The plotter polls the Queue each loop for new plot requests that sent from the RN and plots them.

Class functions:

loadInitParameters(file.xml)- load initial node locations from predefined xml file, and plots them.

placeNode(coors) - placing node manually by mouse clicking

arrows({CID1,CID2,c,txt}, {CID5, CID7,c,txt},{ CID18, CID21,c,txt},…)-   
crates arrows with specified color and text

circles({CID1,coor, color,dot\_c},{CID2,coor,color,dot\_c},…)-  
modified the nodes location

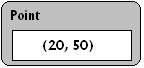
textBox({CIDobject})- shows nodes info in a text box.

Plotter objects and graphics

Node- different outline colors indicates different states  
 (black- off+not seeking, orange- seeking, green- tree connected, blue- object GUI selected)  
   
 the inner circle color indicates the user load  
 (green: <4, orange 5-10 users, red: >10 users, empty: not allow UE connections, blue- object GUI selected)

Root Node (RN)

Connection- indicates UE-eNB connection/request   
 (black- tree connection, cyan- request)



TextBox- displays general info and info about a pressed node.

### Test bench xml file

Xml file that contains initial constant parameters for the simulation to generate different scenarios.

Parameters:

Number of nodes, Max number of allowed connection to a node, The root CID, Connectivity threshold.

The plotter loads the file on startup.

### Scan\_mat xml file

Xml file that gathers all the information about the connections between the nodes, such as distance and connection state.

The RN updates the file with new data each time period or tree topology changes.

### XLS-Visual basic

Excel table that displays the scan\_mat data in real time by macro script.

## Algorithm and Simulation

### Algorithm basic assumptions

To make things easier we'll leave the application part to the future and make now some initial assumptions:

1. A node knows if other nodes are "tree connected" and their "tree\_rate" parameter.
2. All the nodes know about the existence of other nodes around them and the "scan\_mat" file is also known by all the nodes.
3. Nodes can connect to a specific node (not always to the one with the best reception)
4. Routes and other layer 3 configurations configured automatically.

### System Behavior

There are three ways to place nodes on the plot:

1. Xml file- Specific scenario from a Test bench xml file
2. Automatic- the plotter generates the nodes locations
3. Manual- the user places nodes by mouse clicking

The location of a placed node can be changed by clicking on it, choosing the right option and replacing it with the mouse.

A current distributed node plot can be export to an xml file and import in other time for certain condition tests.

The simulation starts and the MST algorithm takes action.

Each node behaves as a separated process (thread).

The next algorithm steps are not ordered specifically by chronologic order, some of them can be take part simultaneously.  
  
**Note: connection will be allowed to establish only if the reception signal is above the preconfigured threshold.**

#### Single node algorithm steps

If a node is not connected to any other node, it will take these steps:

1. After startup the node will transmit and open itself to accept UE connections.
2. It will scan the area around it non-stop, if no tree is found it will connect to the node with the best reception signal,  
   if trees are found it will connect to the closest node of the tree with the highest "tree rate" value.
3. From now on the node is a "tree connected node".

#### Tree connected node algorithm steps

If a node is a "tree connected node" (connected to a tree), it will take these steps:

1. Each node rates itself and checks if it should be the RN of the tree every time that the number of nodes that are connected to it is changes.  
   it will compare its own "node rate" to the current RN "node rate" and if it has a greater "node rate" it will become the new tree RN  
   (the other nodes will be informed that there is a new RN)  
   Each tree declares one node as a RN  
     
   **Note:** nodeRate= (no. of nodes connected to it)\*1000+CID
2. Each tree rates itself (tree rate) and starts to behave logically like a one transmitting node.
3. If another tree is in the area with grater tree rate value, the closest node from the lower "tree rate" connects to the grater tree.

This process will continue till the lower rate tree will completely merged to the greater one.

### Simulation illustration example

1. nodes are booting up and starting to request connections to the closest nodes
2. trees are establishing and RNs declared
3. The minor tree starts to merge with the greater one
4. The two trees are merged to one and one RN is declared.

## Plotter GUI interface

TBD