Using the Contiki Cooja Simulator

Anuj Sehgal

Computer Science, Jacobs University Bremen
Campus Ring 1, 28759 Bremen, Germany
s.anuj@jacobs-university.de

Abstract

Setting up large networks on physical nodes can pose a challenge, as such, using a simulator to develop and test systems can be quite useful. Simulators can allow rapid prototyping and testing on large networks, while also being able to execute tasks at faster than real-time speeds. Cooja is a simulator provided by Contiki, which unlike most simulators also allows real hardware platforms to be emulated. In this tutorial you will be learning to use the Cooja Simulator to setup a basic RPL network.

1. Introduction

This document aims to show the very basics of using the Contiki Cooja simulator and setting up an RPL network within it. It assumes that you have already got a program written to simulate, or that you will be using the RPL examples provided with Contiki 2.6. We will simulate programs working on a TelosB (TMote Sky).

2. Compile Program

For the tutorial, we will setup an RPL network using TelosB motes in the Cooja simulator. Since Contiki 2.6 has a stable version of RPL provided, it is recommended that you download and work with this version within the virtual machine provided to you. Once downloaded, you need to compile the border router and a normal UDP application.

2.1. Border Router

The RPL border router is used in order to interface a regular IP network with an RPL 6LoWPAN network. This is similar to the bridge that was previously used, except that it also runs an RPL network. The border router is located in the `contiki-2.6/examples/ipv6/rpl-border-router` folder. You must compile it using the following commands:

```
1 cd contiki-2.6/examples/ipv6/rpl-border-router
2 make TARGET=sky
```

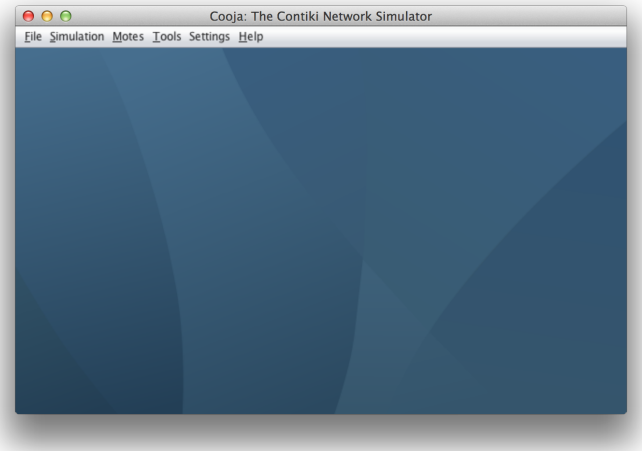


Figure 1: The Cooja desktop interface.

This leads to the `border-router.sky` executable, for the TelosB platform, being created. This executable will be used in Cooja to create a bridge with the RPL network.

2.2. UDP Server

Besides the border router, it is important to have simple UDP nodes within the simulation as well, so that a regular RPL network will be formed. We will be using the RPL UDP Server code to setup the non-border router nodes in the network. Sample code provided by Contiki for this can be found in the `contiki-2.6/examples/ipv6/rpl-udp` folder. You must compile it using the following commands:

```
cd contiki-2.6/examples/ipv6/rpl-border-router
make TARGET=sky
```

We will use the `udp-server.sky` executable in the Cooja simulator.

3. Cooja Simulator

The Cooja simulator is located in the `contiki-2.6/tools/cooja` folder. It uses a combination of Java code for the front-end interface and platform specific emulators to carry out the simulations.

3.1. Starting Cooja

You can start the Cooja simulator with the following commands:

```
cd contiki-2.6/tools/cooja
ant run
```

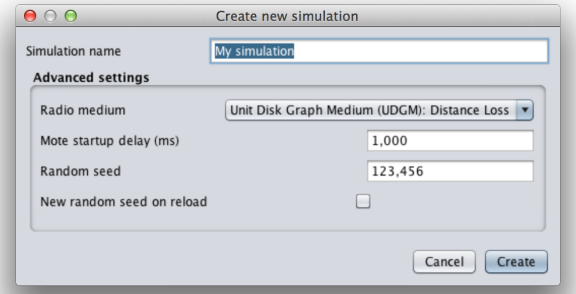


Figure 2: Create new simulation interface.

You will see lots of text go past as the Cooja environment is compiled, built and started. Eventually you will now see the Cooja desktop interface, as shown in Figure 1.

3.2. Creating A Simulation

Firstly you will need to create a new simulation. To do this, click "*File > New Simulation*". This will present a dialog box, shown in Figure 2.

Give your simulation a more suitable title, and then select the radio medium that best suits your simulation type. For most simulations, Unit Disk Graph Medium (UDGM) is quite suitable. Turning radio simulation off by selecting "*No Radio Traffic*", when not needed saves CPU time and makes the simulation run quicker. Random start-up delays the booting of each mote simulated randomly so they don't all start exactly at the same time. The main random seed is a seed for the random number generator. You can tick the box at the end to get a random seed. When you're ready click "Create".

You will now see the Cooja desktop once more, but with the simulation environment presented, as shown in Figure 3.

3.3. The Simulation Interface

The simulation interface, shown in Figure 3, consists of five windows. The Network window shows the physical layout of the network, i.e. you will be able to physically place motes here and move them around, as needed, in order to form the topology and layout you are interested in. The Simulation Control window lets you start, stop and reload the simulation. It also lets you control the rate at which the simulation proceeds.

The Mote Output window shows any serial output generated by all the motes, i.e. the output from the `printf` command. You may filter the output shown based on the string you enter into the "Filter" field. For example, if you wish to filter the output such that it only shows output from mote 2, then you can enter "ID:2" in this field.

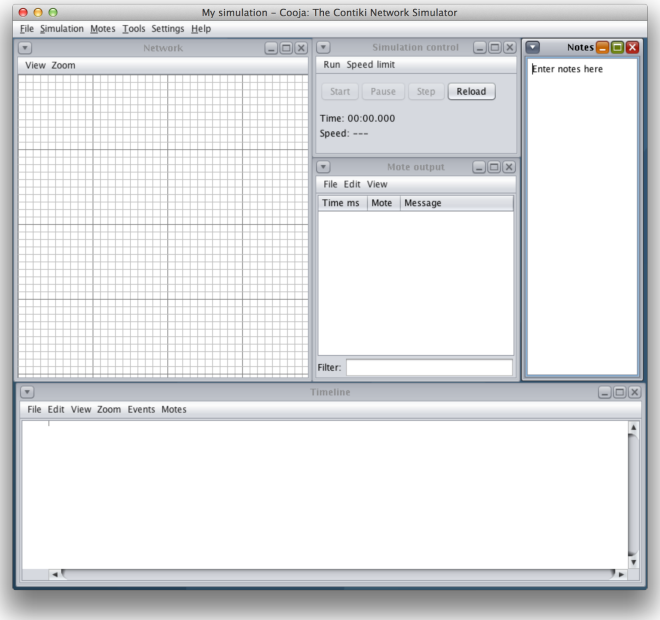


Figure 3: The simulation interface.

The Timeline window shows events that occur on each mote over the timeline of simulation. These events can be radio traffic, LED activity or anything else. The Notes window can be used to take temporary notes in the simulation.

3.4. Setting Mote Types

The next stage is to set the mote types. In Cooja, you create a list of mote types, define their parameters and assign their program code/binary. You then, as a seperate task, create instances of them to simulate. Here we address the first part: creating the mote types.

Click "*Motes* > *Add Motes* > *Create New Mote Type* > *Sky Mote*". Give the mote type a useful description. Let the first mote type be the border router, so we name this mote Border Router, as shown in Figure 4.

In the "Contiki Process / Firmware" field, you should specify your source file (the .c file) or the binary file (the .sky file). If you specify the binary file, you won't see any compile instructions (binary files are the result of compilation – there is no need to compile). If you specify the source code, then the compile commands field becomes active, and you can specify specific instructions for compilation. In this case, since we have already compiled our binaries before, we will just ue the Browse button to select the `border-router.sky` file.

If you have specified source code, you need to press Compile before you can create the mote type. You can see the compilation output/results in the

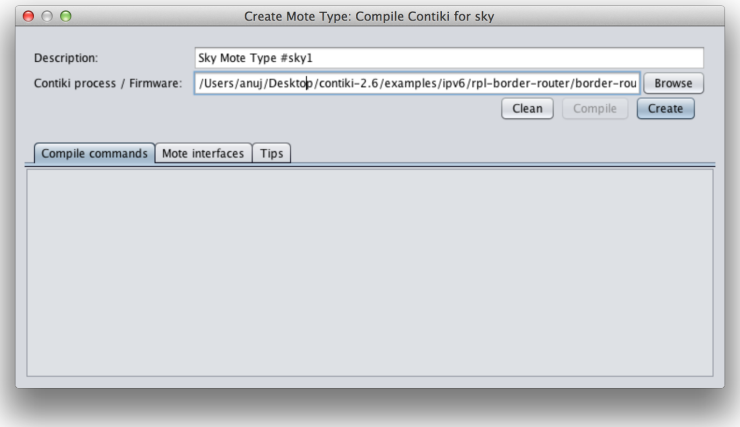


Figure 4: The create mote type interface.

compilation output tab. If successful, the Create button becomes available. In our case, we are using a pre-compiled binary, as such, the Create button is already available. You may now use this button to create a mote of the Border Router type. If you are presented with the "*Add Motes*" window, please click "*Do not add motes*" for the moment.

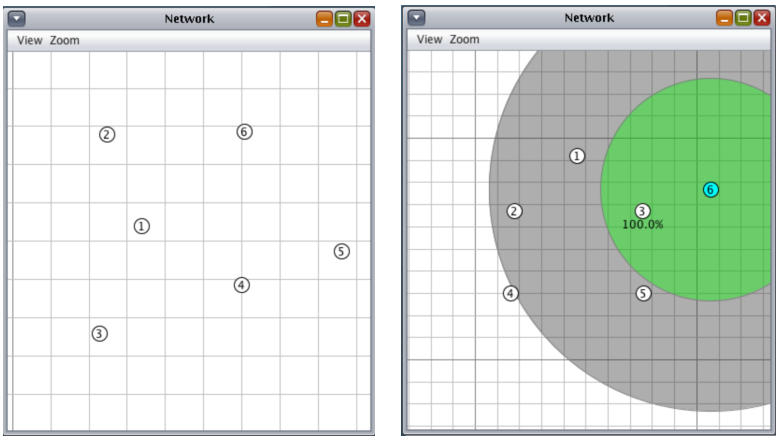
Similarly, you may create the UDP Server mote by using the `udp-server.sky` file we compiled before.

3.5. Adding Motes

Now that you have create the Border Router and UDP Server type motes, we can start adding physical nodes to the simulation. The "*Add motes*" dialogue allows you to set the number and configuration of these new motes. This can be accessed by clicking "*Motes* > *Add Motes*". Add one mote of the type Border Router and five mote of type UDP Server. You may use random positioning for adding the nodes.

Once you do this, you will notice that a total of six randomly placed nodes will appear in the Network window. One possible random arrangement can be seen in Figure 5(a). Amongst these, node 1 is the Border Router and the rest are nodes which will execute the UDP Server code.

You can interact with each of the motes by clicking and dragging them around. You may re-organize the layout of the nodes by dragging them around, such that they will form a topology that you are interested in. When you click on a node, it also shows the radio environment in green and grey color. The green circle represents the area within which the signal is successfully received by other nodes and grey represents the area where radio interference from the current node exists. An example of this radio environment, and a new layout



(a) (b)

Figure 5: The network window.

such that only a maximum of two nodes are within range of each node, can be seen in Figure 5(b).

3.6. Bridging

Even though the network layout is now setup, before starting the simulation a bridge with the RPL network must be created. Since node 1 is the Border Router, the bridge will be created with this node. Right click mote 1 and then select "*More tools for Border Router > Serial Socket (SERVER)*". This creates a serial port on the simulated node 1, which is accessible via UDP port number 60001 on the local machine.

You can now start the simulation by clicking the "*Start*" button in the Simulation Control window.

Now, in a terminal window, enter the following commands in order to setup the bridge:

```
1 cd contiki-2.6/tools
2 make tunslip6
3 sudo ./tunslip6 -a 127.0.0.1 aaaa::1/64
```

This will now setup a bridge into the RPL network, via node 1 of the simulation. The prefix of all nodes in the network will be `aaaa::/64`. Once you enter this command, you will see output that looks similar to the following:

```
1 slip connected to ``127.0.0.1:60001''
2 opened tun device ``/dev/tun0''
3 ifconfig tun0 inet `hostname` up
4 ifconfig tun0 add aaaa::1/64
5 ifconfig tun0 add fe80::0:0:0:1/64
6 ifconfig tun0
7
```

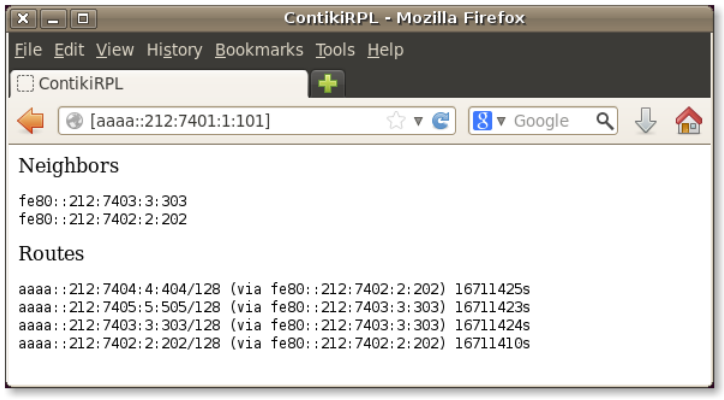


Figure 6: The RPL route status, as visible to the border router.

```
8  tun0    Link encap:UNSPEC  HWaddr
       00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
9          inet addr:127.0.1.1  P-t-P:127.0.1.1  Mask:255.255.255.255
10         inet6 addr: fe80::1/64 Scope:Link
11         inet6 addr: aaaa::1/64 Scope:Global
12
13 *** Address:aaaa::1 => aaaa:0000:0000:0000
14 Got configuration message of type P
15 Setting prefix aaaa::
16 Server IPv6 addresses:
17  aaaa::212:7401:1:101
18  fe80::212:7401:1:101
```

This output confirms that the bridging is setup and that the border router has the address `aaaa::212:7401:1:101` setup. We can verify this by pinging this address using the `ping6` command.

You can also see the current RPL network status by going to the web interface located at the border router, as shown in Figure 6. The IP address for node 2 is `aaaa::212:7402:2:202` and so on.

4. Conclusion

You now know how to work with the Contiki Cooja simulator.