

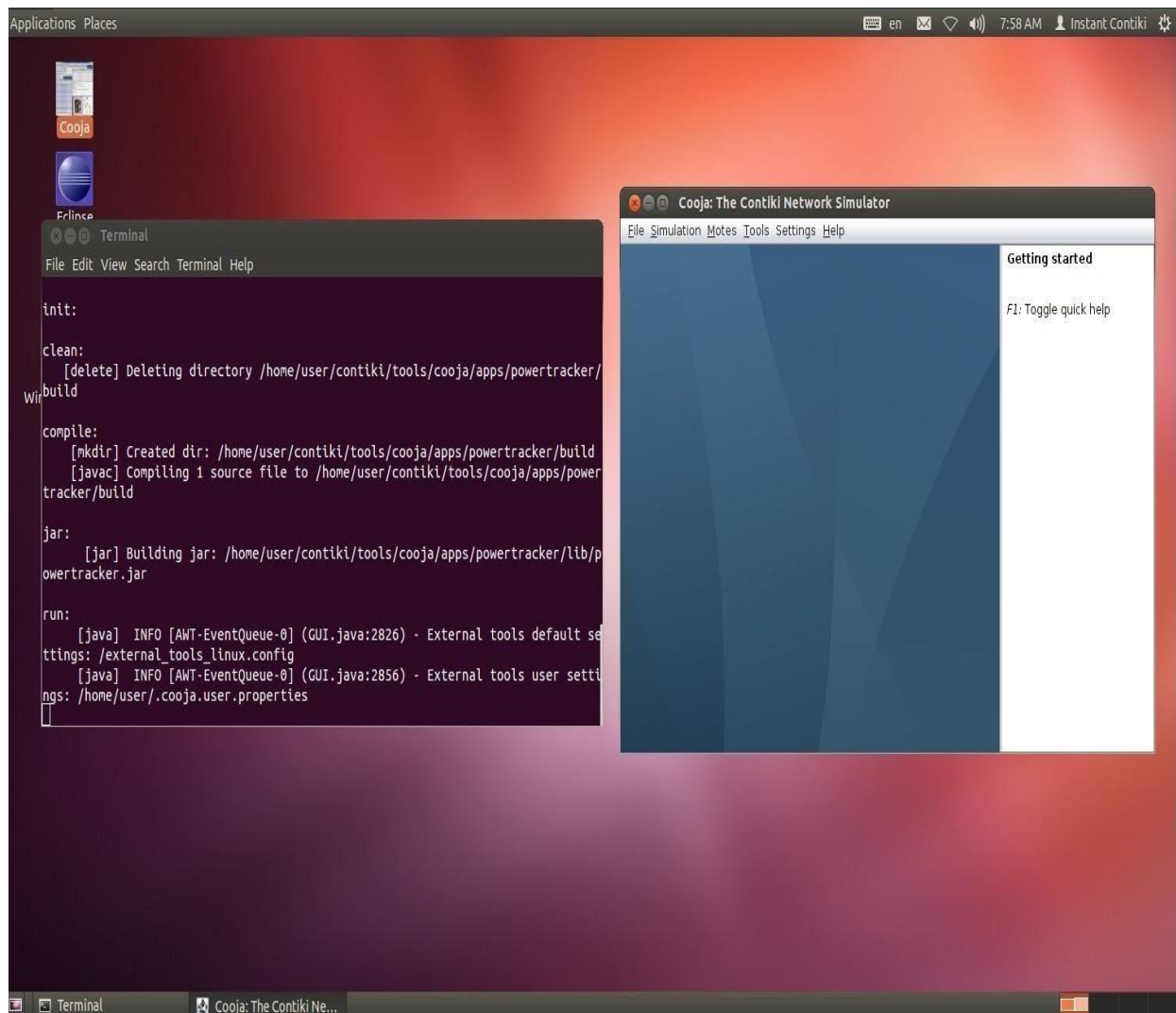
CS 501 – Internet of Things

Assignment-4

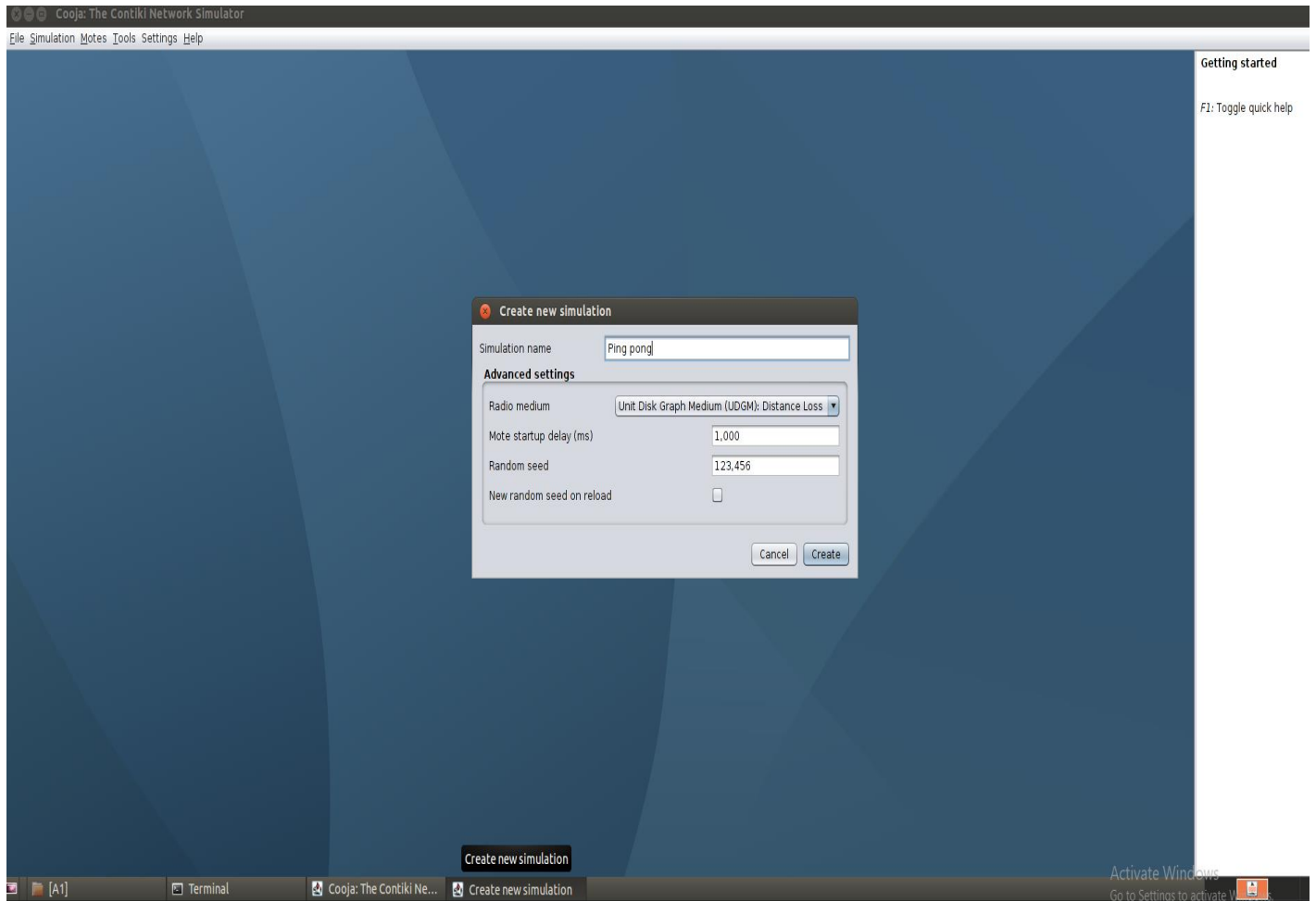
Group Members

Student Name	Student Id	Student Email ID
Lohitha Yalavarthi	002289255	lyalavarthi20@ubishops.ca
Nitish Kumar Pilla	002286814	npilla20@ubishops.ca
Bhargav Movva	002292699	BMOVVA21@ubishops.ca

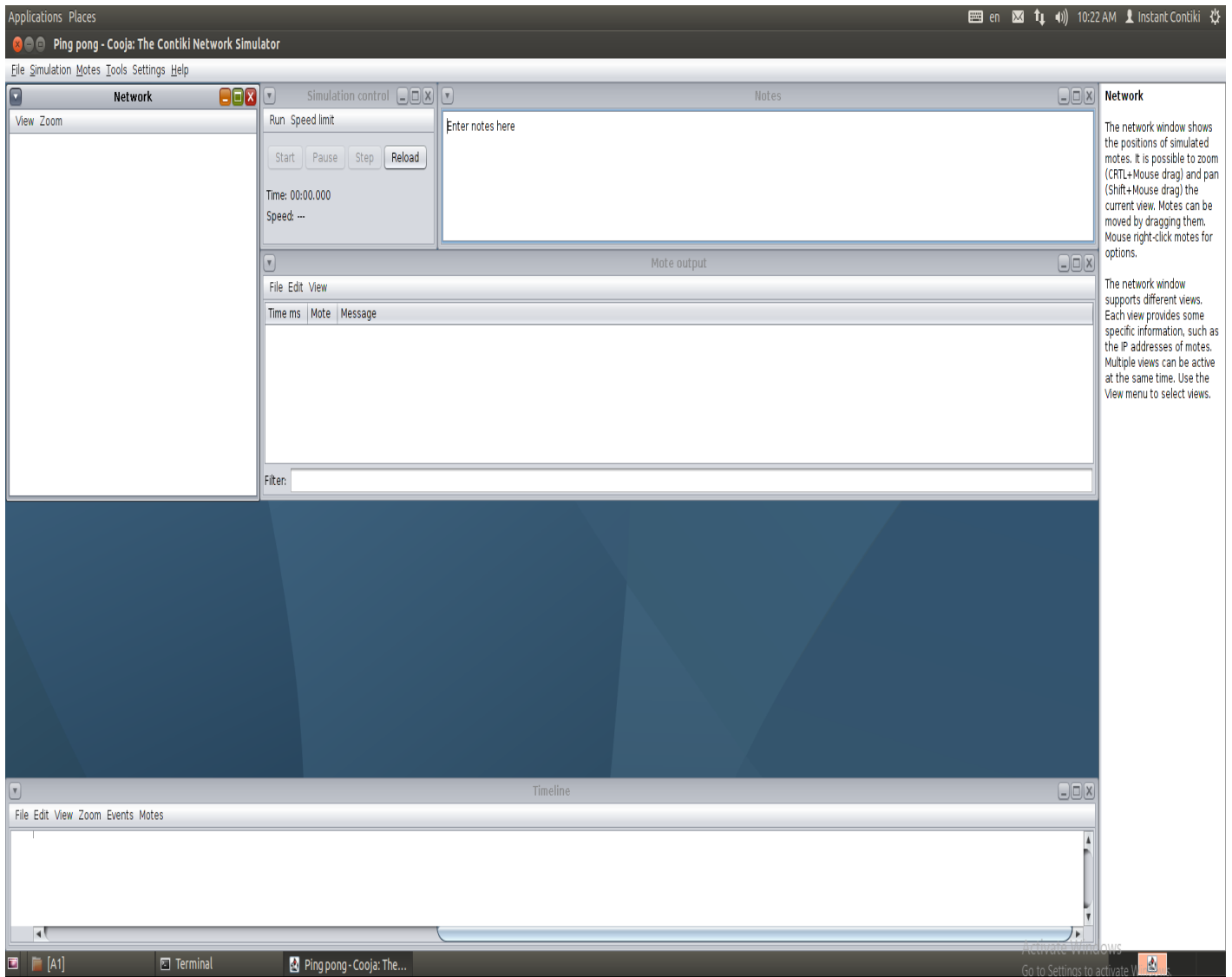
1. Click on Cooja Desktop Icon and it opens the cooja: The Contiki Network Simulator



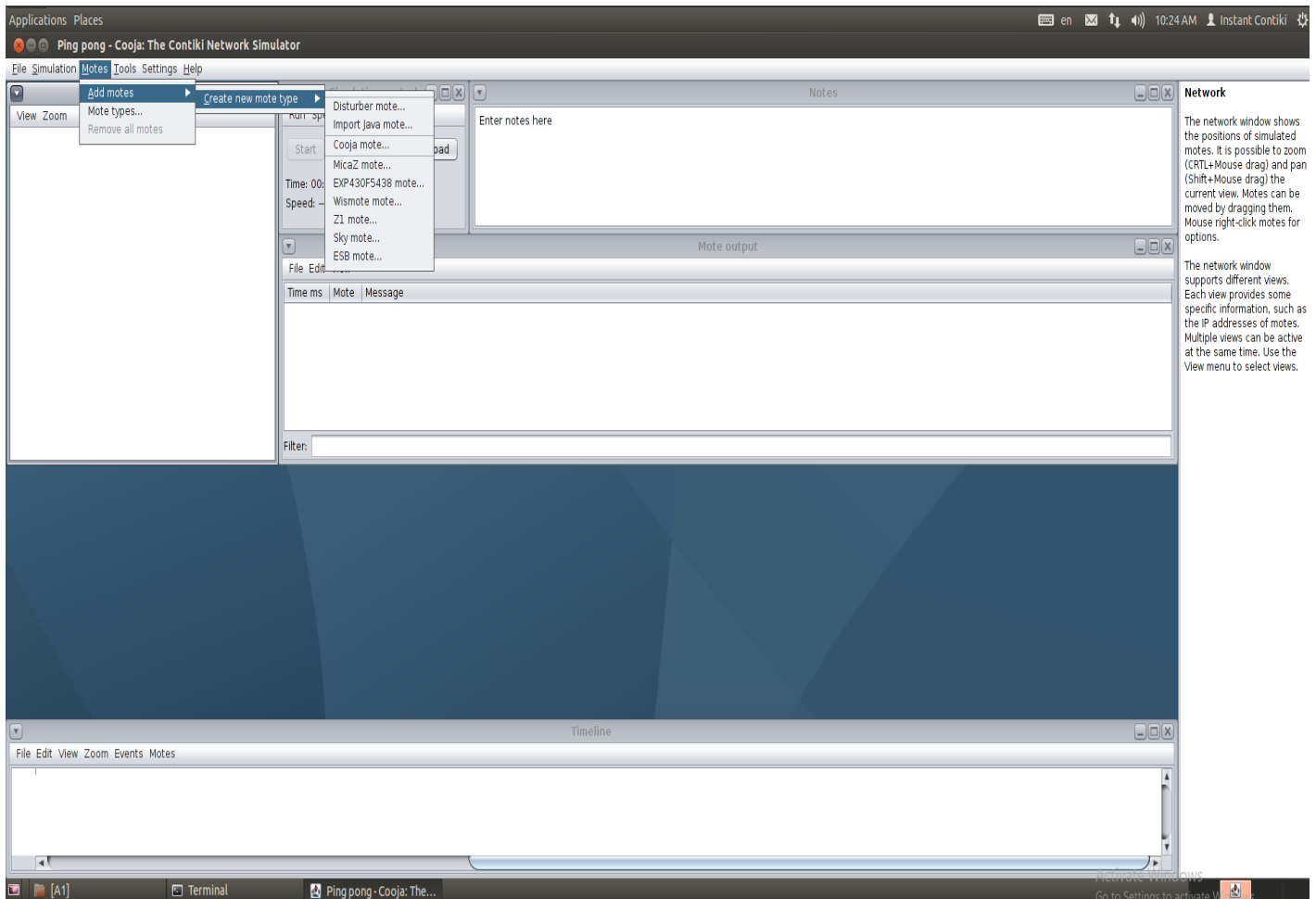
2. Click the new simulation under File menu and it opens the create new simulation dialog and name the simulation as **Ping pong** as shown in the screenshot and a new simulation gets created by clicking on the create button.



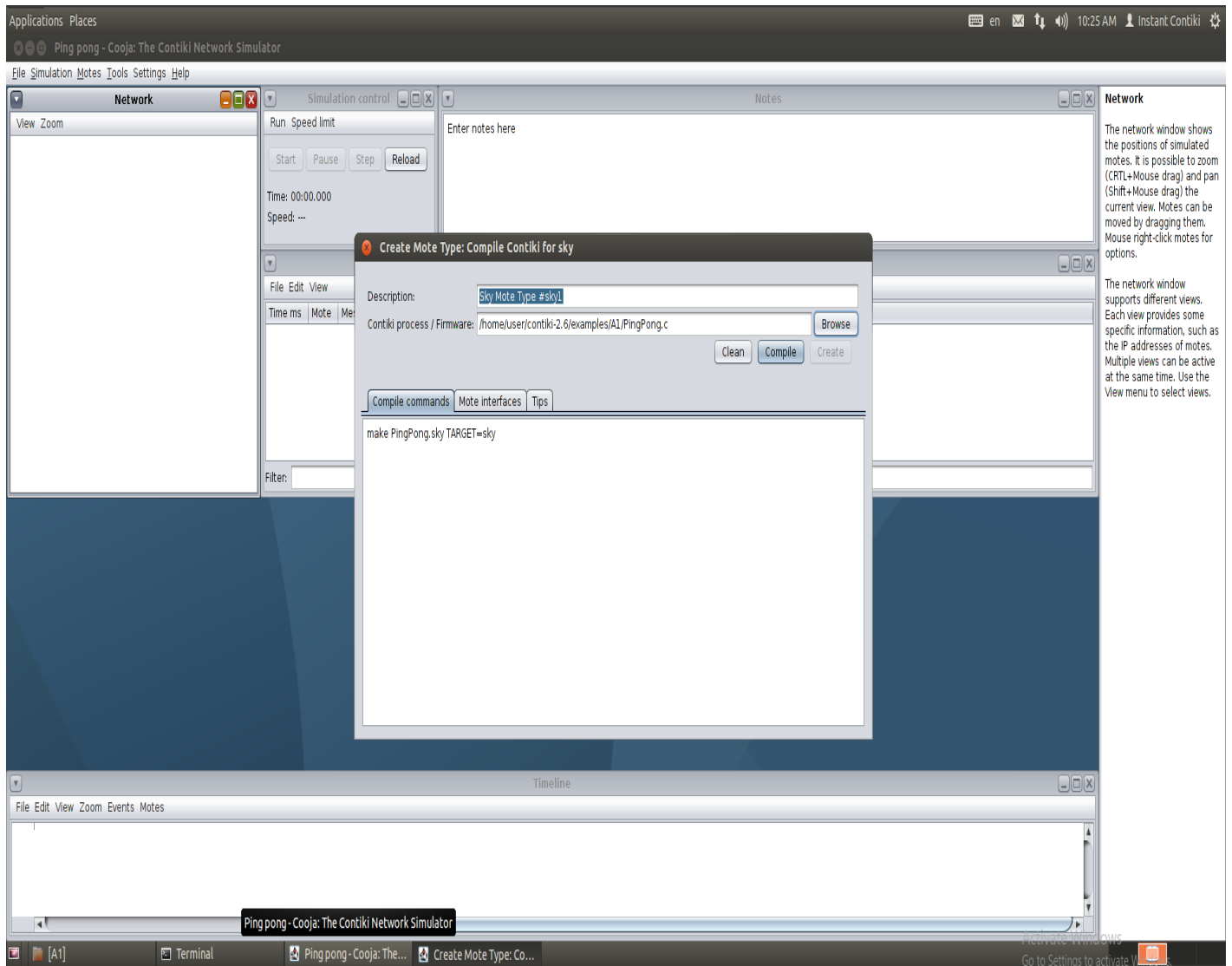
3. It brings up the new simulator window (**Ping Pong-Cooja: The Contiki Network Simulator**) which has Network and motes output and Notes and Timeline and Simulation control windows.



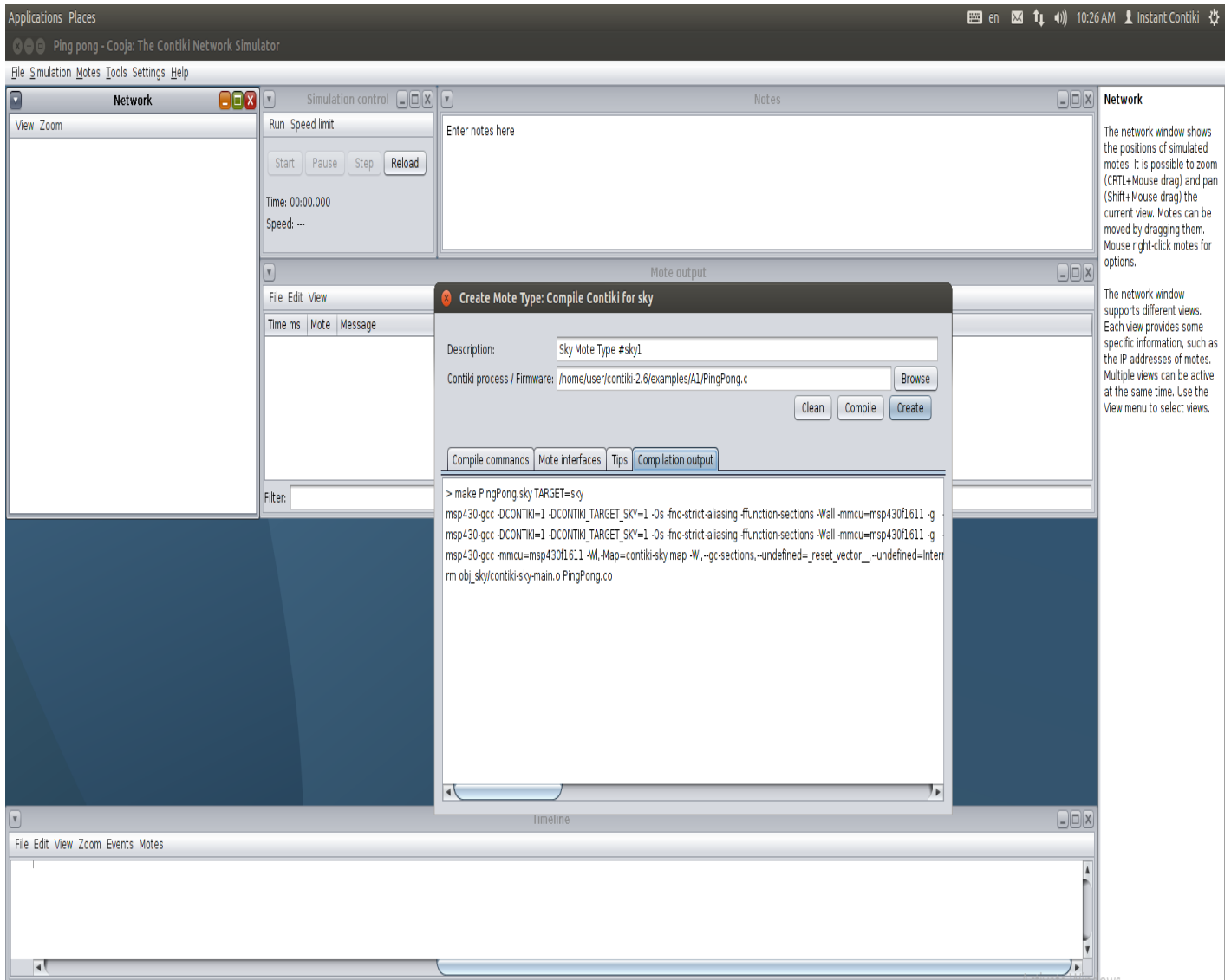
4. Click on Motes option in the simulator menu and select Add Motes->Create new moto type-> Sky mote to create sky mote type.



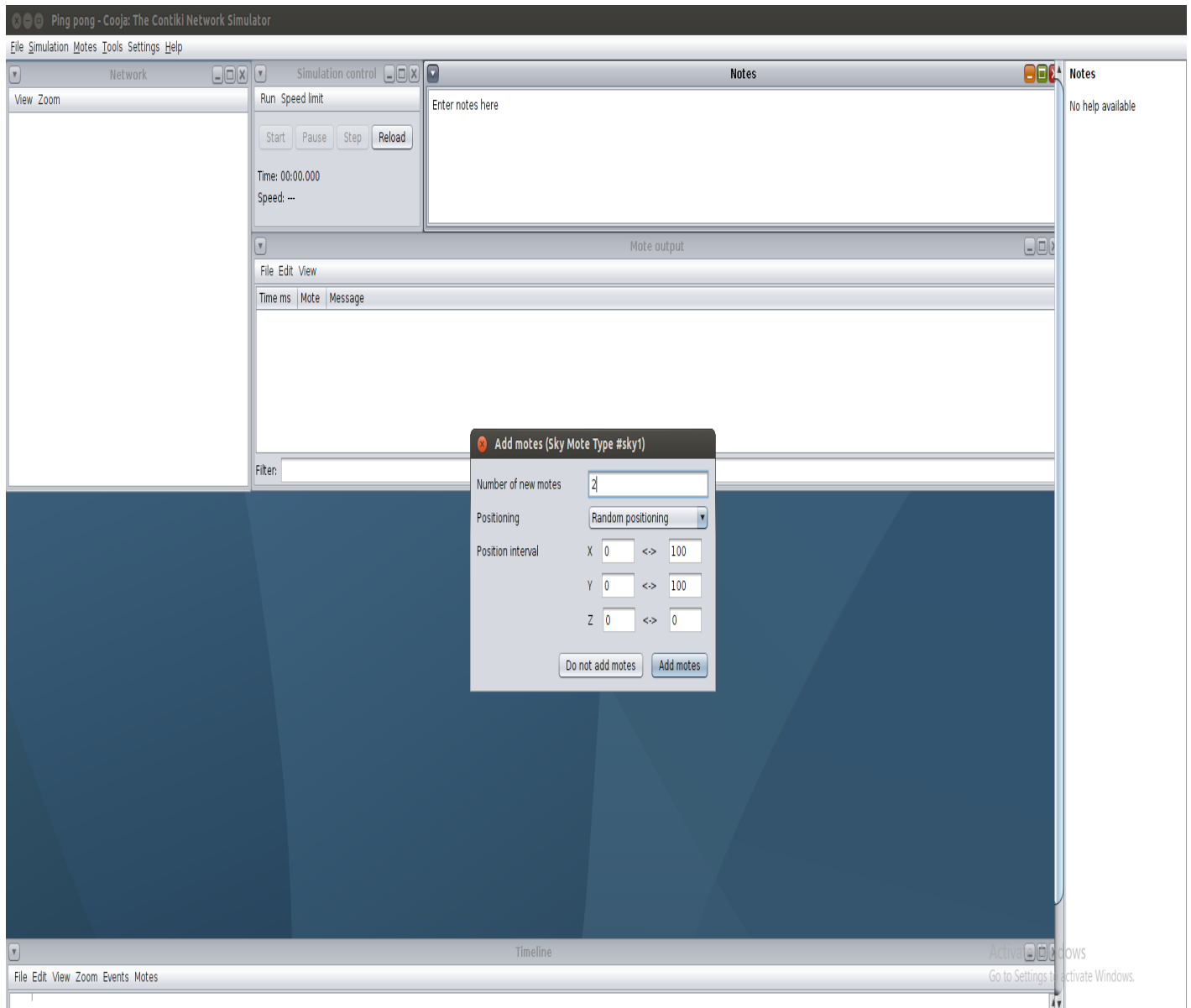
5. After selecting the above options opens **Create Mote Type : Compile Contiki for sky** dialog where Description name and Contiki process/Firmware is given.



6. Click the compile button and the compilation results shows in the compilation output tab.

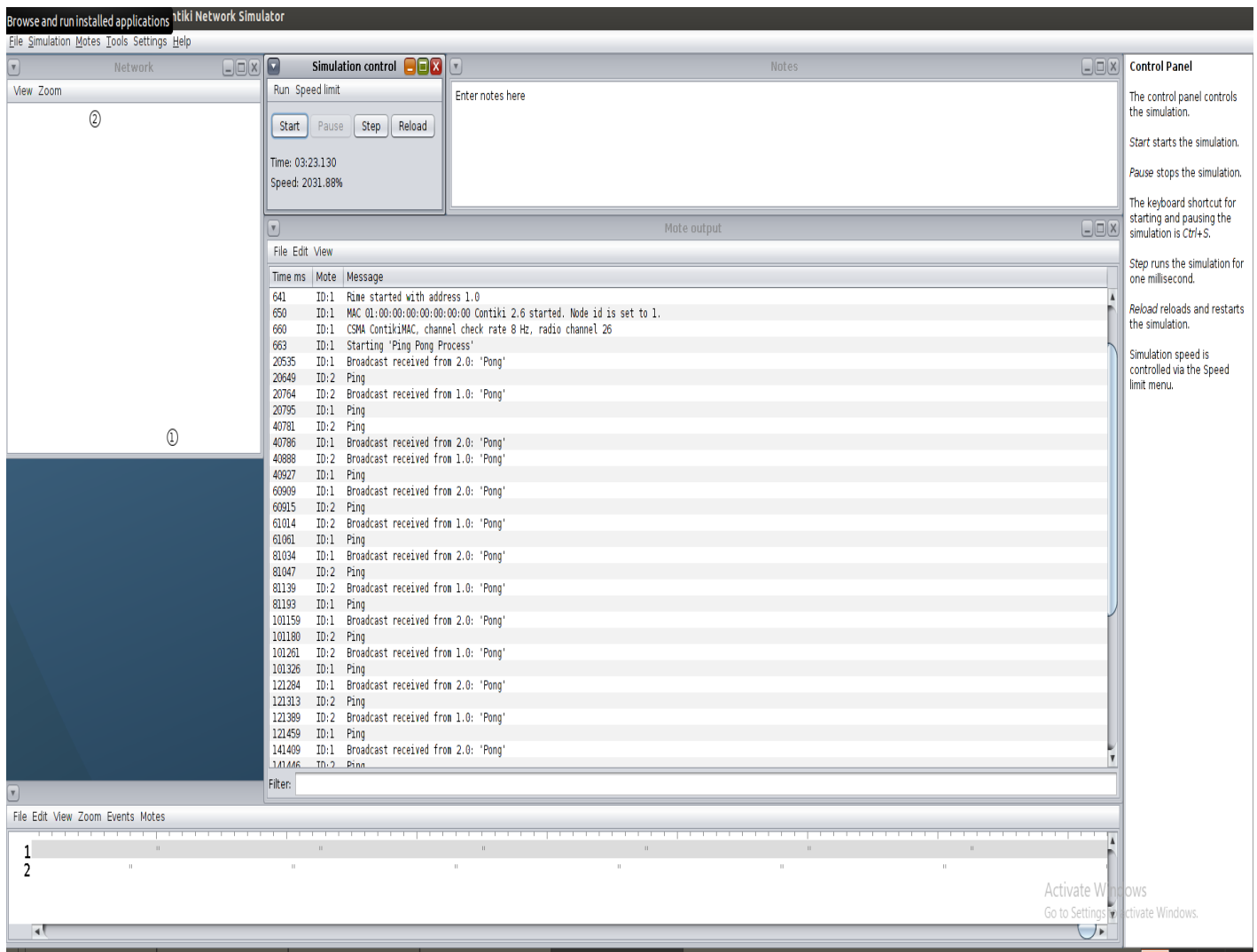


7. After successful compilation, clicking on the create button opens a dialog to enter the no of motes. no of motes entered here are 2 as shown in the screenshot.



8. Click the start button in the simulation control window to start the simulation and the compilation results are seen in the mote output and the timeline window.

Here we have added up to 2 motes only and we can observe that pong response is being sent from nodes.



9. Now we will add another 4 motes and reload the simulation.

By adding extra motes, we can clearly see that the pong response is received only from some nodes. From remaining nodes there is no response

The screenshot displays the Cooja: The Contiki Network Simulator interface. The main window is titled "ping pong - Cooja: The Contiki Network Simulator". It features a menu bar (File, Simulation, Notes, Tools, Settings, Help) and a toolbar with icons for Network, Simulation control, and Notes.

The Network view shows a diagram with 4 motes labeled 1, 2, 3, and 4. Mote 3 is highlighted in blue. The Simulation control panel includes buttons for Start, Pause, Step, and Reload, along with a Time display (03:10.942) and a Speed limit (---). The Notes panel is empty, with a placeholder text "Enter notes here".

The Mote output panel displays a log of network events. The log is organized into columns: Time ms, Mote, and Message. The messages show the simulation's progress, including the start of the 'Ping Pong Process', the initialization of the MAC layer, and the receipt of broadcast messages from various motes.

The Log Listener panel on the right provides instructions on how to filter logs using regular expressions. It includes a list of filter examples:

- logs containing the string 'Hello'
- logs starting with 'Contiki'
- logs starting either a C or an R
- logs ending with 'Hello'
- logs from motes 2 to 5
- logs from motes 2 to 5 starting with 'Contiki'

The bottom of the interface shows a timeline with a scale from 0 to 1000 ms, and a list of events (1, 2, 3, 4) corresponding to the motes in the network view.

The broadcast protocol is inefficient because:

A)

From the above simulations, we can clearly see that adding a greater number of nodes in broadcast protocol caused some irregular responses or some nodes are not receiving proper responses to pings.

This might be the reason for broadcast protocol being inefficient.

It is also inefficient because a lot of bandwidth is being wasted in the broadcast protocol.

How could the protocol be made more efficient?

A)

We can place time delay before receiving pong response from nodes to make it more efficient.

If we add more nodes in the network, and the nodes in the network(left side) are dispersed far away from each other, Then the nodes that are located far away from other nodes are not responding with other nodes and not broadcasting properly. For that we have to make the nodes in the network close to each other (i.e changing the positions of nodes close to each other). Then the broadcasting of each node is being executed properly.

Here the notes in the network are placed close to each other. Hence we can see that broadcast is received for all nodes correctly.

The screenshot displays the 'ping pong - Coolidge: The Contiki Network Simulator' window. The interface is divided into several panels:

- Network:** A diagram showing six nodes (1-6) arranged in a circular pattern.
- Simulation control:** Includes buttons for 'Run', 'Speed limit', 'Start', 'Pause', 'Step', and 'Reload'. The current time is 04:16.802.
- Notes:** A text area for entering notes.
- Mote output:** A log window showing a list of messages received by various nodes. The messages are categorized by ID and type (Broadcast received from, Ping).
- Log Listener:** A panel on the right side providing instructions on how to filter logs using regular expressions.

Mote output log:

Time ms	Mote	Message
237909	ID:1	Broadcast received from 3.0: 'Pong'
237915	ID:3	Ping
238139	ID:2	Broadcast received from 6.0: 'Pong'
238160	ID:1	Broadcast received from 6.0: 'Pong'
238178	ID:4	Broadcast received from 6.0: 'Pong'
238208	ID:5	Broadcast received from 6.0: 'Pong'
238249	ID:3	Broadcast received from 6.0: 'Pong'
238258	ID:6	Ping
238374	ID:3	Broadcast received from 4.0: 'Pong'
238390	ID:2	Broadcast received from 4.0: 'Pong'
238410	ID:1	Broadcast received from 4.0: 'Pong'
238459	ID:5	Broadcast received from 4.0: 'Pong'
238467	ID:6	Broadcast received from 4.0: 'Pong'
238470	ID:4	Ping
238639	ID:2	Broadcast received from 5.0: 'Pong'
238660	ID:1	Broadcast received from 5.0: 'Pong'
238678	ID:4	Broadcast received from 5.0: 'Pong'
238717	ID:6	Broadcast received from 5.0: 'Pong'
238747	ID:3	Broadcast received from 5.0: 'Pong'
238751	ID:5	Ping
241998	ID:3	Broadcast received from 2.0: 'Pong'
242035	ID:1	Broadcast received from 2.0: 'Pong'
242053	ID:4	Broadcast received from 2.0: 'Pong'
242083	ID:5	Broadcast received from 2.0: 'Pong'
242091	ID:6	Broadcast received from 2.0: 'Pong'
242110	ID:2	Ping
242139	ID:2	Broadcast received from 1.0: 'Pong'
242178	ID:4	Broadcast received from 1.0: 'Pong'
242209	ID:5	Broadcast received from 1.0: 'Pong'
242217	ID:6	Broadcast received from 1.0: 'Pong'
242247	ID:3	Broadcast received from 1.0: 'Pong'
242256	ID:1	Ping

Log Listener instructions:

- Lists to log output from all simulated motes. Right-click the main area for a popup menu with more options.
- You may filter shown logs by entering regular expressions in the bottom text field. Filtering is performed on both the Mote and the Data columns.
- Filter examples:**
 - ^Contiki: logs containing the string 'Contiki'
 - ^Hello: logs starting either a C or an R
 - ^ID:[2-5]: logs from motes 2 to 5
 - ^ID:[2-5] Contiki: logs from motes 2 to 5 starting with 'Contiki'

=====The End=====