Reproducing Paper Experiments

This document describes how to reproduce the experiments in the paper "Um Simulador de Protocolos e Algoritmos de Redes Escrito em Python".

Ping Pong

Firstly we will reproduce the ping pong experiment. The ping pong experiment consists of two nodes that send messages to each other. The messages are sent with a random color in the content that are used by the receiver to color themselves.

- If you haven't already done the #Getting Started tutorial in the README, do so before reproducing the experiment.

1. Navigate in your terminal to the SNAPPY root folder.
2. Run the following command to start the simulator:

```
$ source activate mobenv
```

```
$ python manage.py runserver
```

3. Open your browser and go to http://localhost:8000/mobsinet/graph/.
4. Open the configurations menu by clicking in the "Show/Hide configurations" button if it is not already open.
5. Select project "Sample1" in the project selection dropdown.
6. Check if the rest of the configurations are set as follows:
    - Node: sample1:pingpong_node
    - Number of Nodes: 2
    - Size of nodes: 10 # You can change this value to see the nodes better
    - Dimension X: 1000
    - Dimension Y: 1000
    - Distribution Model: linear_dist
    - Orientation: horizontal
    - Line position: 0
    - Number of nodes (for section Distribution Model Parameters): 2
    - Mobility Model: random_walk
    - Speed Range: 0,20
    - Direction Range (radians): 0,6.283185307179586
    - Prioritize Speed: No
    - Travel Time: 70
    - Connectivity Model: qudg_connectivity
    - Max Radius: 1000
    - Min Radius: 800
    - Big Radius Probability: 0,75
    - Reliability Model: reliable_delivery
    - Interference Model: probability_interference
    - Intensity: 5

- Message Transmition Model: random_time
- Random Transmition Min Time: 2
- Random Transmition Max Time: 4

7. Click on "Submit" button at the end of form.
8. Click on "Initialize" button to reset simulation variables and dispose the nodes in the scenario.
9. Set the number of rounds to 60.
10. Set the refresh rate to 15.
11. Set the simulation fps to 60.
12. Click on "Run" button to start the simulation. You should see the nodes moving in the scenario and sending messages to each other. The messages are colored and the receiver nodes change their color to the color of the message. You can see when a message is droped or successfully delivered in the integrated console. When the nodes loses their connection or reconnect, a message is displayed in the logs console.
13. To turn the IDs of the nodes always visible, click and hold on the "Show IDs" button move the mouse out the button and release the mouse.
14. To turn the arrows aways visible, use the same procedure as in the step 13 but with the "Show Arrows" button.
15. To stop the simulation, click on the "Stop" button.
16. Input the node IDs in the fields "Node 1" and "Node 2" and click on the "Distance" button to see the euclidean distance between the two nodes.

Shortest Path + Node2Vec

Now we will reproduce the shortest path with Node2Vec experiments. The shortest path experiment consists of finding the shortest path between two nodes in the scenario. The Node2Vec experiment consists of running the Node2Vec algorithm in the current scenario.

- If you haven't already done the #Getting Started tutorial in the README, do so before reproducing the experiment.

1. Navigate in your terminal to the SNAPPY root folder.
2. Run the following command to start the simulator:

```
$ source activate mobenv
```

```
$ python manage.py runserver
```

3. Open your browser and go to http://localhost:8000/mobsinet/graph/.
4. Open the configurations menu by clicking in the "Show/Hide configurations" button if it is not already open.
5. Select project "Sample4" in the project selection dropdown.
6. Check if the rest of the configurations are set as follows:
   - NACK Messages Enabled: Yes
   - Node: sample5:ping_node

- Number of Nodes: 100
- Size of nodes: 3
- Dimension X: 10000
- Dimension Y: 10000
- Distribution Model: circular_dist
- Number of nodes (for section Distribution Model Parameters): 100
- Midpoint: 5000,5000
- Rotation Direction: anti-clockwise
- Radius: 500
- Mobility Model: random_walk
- Speed Range: 100,120
- Direction Range (radians): 0,6.283185307179586
- Prioritize Speed: No
- Travel Distance: 500
- Connectivity Model: qudg_connectivity
- Max Radius: 500
- Min Radius: 200
- Big Radius Probability: 0,2
- Reliability Model: reliable_delivery
- Interference Model: no_interference
- Message Transmition Model: random_time
- Random Transmition Min Time: 1
- Random Transmition Max Time: 10

7. Click on "Submit" button at the end of form.
8. Click on "Initialize" button to reset simulation variables and dispose the nodes in the scenario.
9. Click in the "Show/Hide add nodes form" button to open the form.
10. Change the follow parameters in the form:
    - Radius (for section Distribution Model Parameters): 1500
    - Max Radius (for section Connectivity Model Parameters): 1000
    - Min Radius (for section Connectivity Model Parameters): 500
11. Click on the "Add to the simulation" button to add the new batch of nodes to the simulation.
12. Go back to the Add Nodes form and change the follow parameters in the form:
    - Radius (for section Distribution Model Parameters): 2500
    - Max Radius (for section Connectivity Model Parameters): 1200
13. Click on the "Add to the simulation" button to add the new batch of nodes to the simulation.
14. Set the number of rounds to 1.
15. Click on "Run" button to start the simulation.
16. Input the node IDs in the fields "Node 1" and "Node 2" and click on the "Shortest path" button to run the shortest path algorithm (in the paper, we have tried with nodes 150 and 300). You should see the shortest path between the two nodes in the graph in red arrows.
17. Input the number of dimensions in the field "Dimensions" and click on the

"Run Node2Vec algorithm" button to run the Node2Vec algorithm (in the paper, we have tried with 4 dimensions). You should see the result of the Node2Vec algorithm in the chart.