WIRELESS NETWORK MOVEMENT SIMULATION in NS2

FACULTY of COMPUTER SCIENCE

Hanin Almutairi, Hawra Alseef and Nada Alsalmi

Investigation of NS-2 Movement File Generation for a Curved Path

Hawra Alseef

Introduction

The Mobile Ad Hoc Network (MANET) is a decentralized wireless network, having no fixed infrastructure or designated routers where mobile nodes communicate with each other via dynamically set up connections between the nodes. The mobile nodes move dynamically and follow tracks that determine the mobility pattern in the network. Mobility of mobile nodes significantly affects the performance of a MANET.

Problem

There are many mobility patterns for mobile ad hoc networks. Indeed, the standard way used in MANET simulation for movement generation for current mobility models for MANET simulation is a movement along a straight line between the current location and a designated destination point. To enhance MANET simulations, a new movement generator will be developed and investigated for generating a special movement pattern among different mobile nodes in NS-2 simulation. Specifically, we will be generating movement files for NS-2 simulation that specify the motion along **curved paths**.

Objective

The main objective of this research is to provide a new tool for modifying the simulation environment by modeling motion in the wireless network simulation in a new way. More precisely, we focus on generating (special) movement files for NS2 simulation that specify the motion along **curved paths**. The new tool will be used for experimenting with different movement patterns and comparing their impact on the delivery ratio.

Possible solutions

There are a few simple approaches in which a curved path can be defined.

1. Moving a node along a **fractal curve**.
2. Moving nodes along a **sinusoidal path**.
3. Moving nodes along a **circular path**.

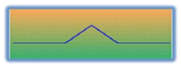
Let us consider the case of implementing the movement along a fractal curve:

As an example we will consider the implementation of the movement along the Von Koch curve [1]. This curve is named after the Swedish mathematician Helge Von Koch who first designed them in 1904. In fact, these curves are amongst the most important fractal objects that allow numerous variations and have inspired many artists to produce amazing pieces of art.

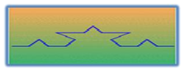
Construction of the Koch Fractal

The construction of this curve is quite simple.

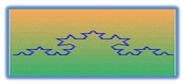
- A straight line is first divided into three equal segments. The middle segment is removed and replaced by two segments having the same length to generate an equilateral triangle. Applying such a 4-sides generator to a straight line leads to the following shape:



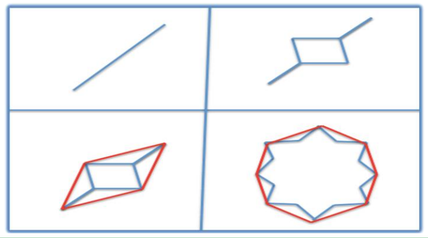
- This process is then repeated for the 4 segments generated at the first iteration, leading to the following drawing in the second iteration of the building process:



- The third iteration already gives a nice picture:



This also applies for implementing an approximation of a circular path:



Implementation

Typically, defining the node movements needs to be done ahead of the NS-2 simulation. In general a curved path can be approximated by a series of short line segments, which determine the final shape of this curve. Therefore, a Java program will be implemented that reads the movement file with random movements generated, for example, by the Setdest utility. Then, as each movement in the movement file is specified by a separate Setdest command, we will replace each one of these Setdest commands, each specifying a movement along a straight line, with **a series** of Setdest commands specifying the movement along a curved path (parametric curve or fractal) as it is illustrated in the coding example below. Once the new movement file is generated the NS-2 simulation can proceed in a standard way.

Example

The sample movement file (scene-3-test) has the following movement statement:

$ns_ at 0.40000000000 "$node_(0) setdest 100.000000 400.00000 1000.02215"

This line specifies that at time 0.40000s, node0 starts to move towards the destination (100,400) at a speed of 1000m/s (this can be one single random movement in a straight line). This single command in the movement file is then replaced by four new commands generating the movement along the path corresponding to the shape of the fractal curve.

References
http://www.tgmdev.be/curvevonkoch.php
http://www.ccs.neu.edu/home/noubir/publications/LNR04.pdf

Cloud Movement GENERATOR for MANETS

Nada Alsalmi
F42j2@unb.ca

Introduction

A mobile Ad Hoc Network (MNAET) is a set of mobile devices that cooperate with each other to perform a certain task. Mobile devices are linked together through wireless connections without infrastructure and can change locations and configure themselves in space. They act as routers and nodes move dynamically and may follow an algorithm that determines the mobility pattern in the network. Mobility of mobile nodes significantly affects the performance of a MANET. There are many mobility patterns for mobile ad hoc networks.

Problem

During the simulation of a MANET network, nodes are free to move around within the network. Previous simulations were performed in a bounded area, which can keep all the nodes in same place. If any node moves far way it will hit the boundary and will then move in another direction but still in the same area with the other nodes. I plan to investigate a scenario to generate node movement path dynamically during the simulation, and without boundaries, while providing means for maintaining all nodes in the same area. When nodes move far way they should be designed to detect where the other nodes are and return back.

Objectives

The main aim is to enhance the ns2 simulation of MANETs by generating scenarios with random movement of nodes in a Mobile Ad hoc Network (MANET) in unlimited geographical area. This means we do not want to use the defined boundaries of the ns2 simulation to confine nodes to a restricted region. More specifically, we would like to allow for the mobile nodes to move randomly, but at the same time we would like to maintain the node density over some region, which means the modes should keep moving and should stay in the same area and not "run away".

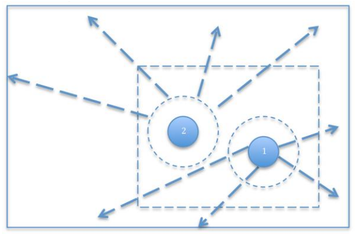


Figure 1:The nodes can move until they hit the boudary define in NS-2.

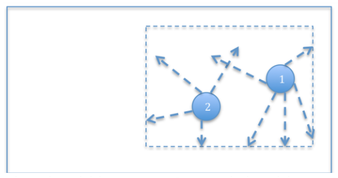


Figure 2: The nodes should keep moving and should stay in the same area and not "run away".

Possible solution

One solution would be to place the nodes at random locations and move them on closed paths e.g. square or circular. Another way would be to place the nodes randomly and then control their node movement dynamically. Each time the new move is to be made by executing setdset command a decision needs to be made to allow node movement in the selected direction. If a node is moving too far from other nodes, the direction needs to be adjusted to return it to other nodes. Evaluation tests will show how random the nodes distribution is during the simulation, using the approach developed in [1].

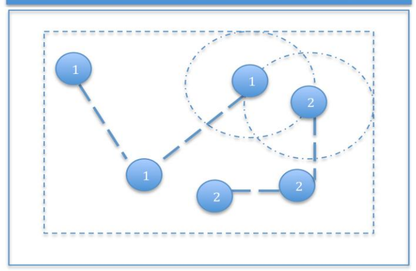
Coding example

Assume that all of the nodes have the same communication range R. The nodes inside the range are called neighbors, and two or more neighbors can communicate. In the example we use nodes being generated randomly at random locations.

```
set xx2 [expr 100+ rand()*200]
set yy2 [expr 150 + rand()*200]
for {set i 2} {$i < 6 } { incr i } {
        $node_($i) set X_ $xx2
        $node_($i) set Y_ 600.0
        $node_($i) set Z_ 0.0
```

Each node has its location, which is simply denoted as N. Having Global Positioning System (GPS) receivers at some fixed nodes can discover the location information. We move the nodes randomly in different directions but in the same n x m area in order to evaluate this scenario. The nodes move at a constant speed of 10m/s during simulation.

```
proc move {} {
global ns_ node_ val
set time 0.5
set now [$ns_ now]
        for { set i 0 } { $i < 1000 } { incr i } {
                for { set j 1 } { $j < 10 } { set j [expr $j * 10]} {
                        set xx2 [expr $j + rand()*1000]
                        set yy2 [expr $j + rand()*1000]
                $ns_ at [expr 30.0 * $i ] "$node_(2) setdest $xx2 $yy2 10.0
}               }       }
```



References
[1] Raid AlGammadi, Investigating Intelligent Node Movement Patterns in Wireless Networks, MCS Report, University of New Brunswick, Fredericton, Canada, November, 2012.
[2] Abdullah Alshehri, Simulating Dynamic Node Movements for MANET in NS-2, MCS Report, University of New Brunswick, Fredericton, Canada, January, 2013.

Mobile Ad-Hoc Networks Dynamic Movements Based on Swarm Intelligence Algorithms

Hanin Almutairi

Introduction

A Mobile Ad-hoc Network (MANET) is a set of self-organizing wireless nodes with no infrastructure or a central node to control the network. Nodes can act as hosts and routers during a transmission. When two nodes are in radio range a connection can be established between them in a peer-to-peer form through single-hop and multi-hop communication. In a MANET nodes are free to move anywhere. In the random movement model nodes might disconnect from the network at any time.

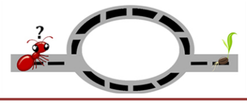
Swarm Intelligence

Swarm Intelligence (SI) is an artificial-intelligence approach to problem solving using algorithms based on the self-organized collective behavior of social insects such as ants, bees, and termites that are each following very basic rules. Insects communicate with each other exchanging the information about locations, paths or food sharing experience about best places they found and the shortest path they can take. SI based on insects' nature provides a basis with which it is possible to explore collective (or distributed) problem solving without centralized control or the provision of a global model.

Swarm Intelligence algorithms can be used in MANET nodes to choose an optimal path for node movement to facilitate a network connection. Nodes implementing an SI algorithm may have a better ability to connect than other types of node [1]. SI nodes move dynamically and are called Agents. The agents follow very simple rules. Although there is no centralized control structure dictating how individual agents should behave, the interactions between SI agents lead to "intelligent" global behavior, unknown to the individual agents.

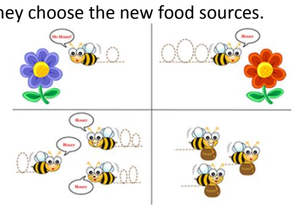
Ant colony optimization algorithms

Ant colony optimization (ACO) is a class of optimization algorithms modeled on the actions of an ant colony in a swarm intelligence approach. ACO methods are useful in problems that need to find paths to some goals. In nature ants leave pheromones trails on the path they took to find food and back again to colony. First ACO algorithms aimed to solve the travelling salesman problem and the goal was to find the shortest round trip between series of cities.



Artificial bee colony algorithm

Artificial bee colony algorithm (ABC) is an optimization algorithm that simulates the honeybee swarm behavior. The bee colony has three types of bees: employed, onlooker and scouts bees. Employed bee has a food source position in her memory and when she discovers a new position she compares the nectar amount of the new one and if it is higher than the previous source the bee memorizes the new source position and forgets the old one. Otherwise she keeps the position of the one in her memory. And by dancing, the employed bees share information with the onlookers about the new source position. Onlookers evaluate this information and based on the nectar amounts they choose the new food sources.



Problems

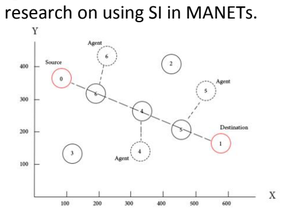
Mobile ad-hoc networks are very flexible and require few resources, which make this kind of networks perfect in the places where the infrastructure is almost impossible to build or maintain. Because of the nature of this network, nodes move randomly without any control which often leads to loss of connections between these nodes. Whenever nodes lose connection it is hard to predict when the next route will be established. The new route is not necessary optimal (shortest) path and long routing paths may lead to transmission delays and power loss. The increase in network size requires more nodes, more time and covering large distances, all of which affects directly the connection path between pairs of nodes.

Objective

The main objective is investing through simulation the performance of different Swarm Intelligence Algorithms and implementing suitable algorithm(s) that allow nodes (Agents) to move in between the source and the destination nodes dynamically in a mobile ad-hoc network environment, finding the optimal path to communicate with each other following simple intelligent rules inspired from the understanding of insect swarm behavior in nature. This optimal path may come from controlling the mobile nodes movement during the simulation and updating their locations to fit the source and destination positions. Designing new routing protocols is not considered in this research thesis. Also, calculation and analysis of different criteria such as throughput, bandwidth, packet delivery rate, delay and speed will be considered in defining the SI algorithm performance.

Network Simulators

NS2 mobility extension for wireless model added some features for the regular nodes to create (Mobile Nodes) so they connected without links – via wireless channels – and move inside the topology freely. These new functionalities allow initializing the position for each mobile node, updating their location during the simulation and control the node movement. NS2 will be used in this research on using SI in MANETs.



Reference
[1] Martin Heinz Roth, "*Termite: A Swarm Intelligent Routing Algorithm For Mobile Wireless Ad-Hoc Networks*", 2005.