A Method for Transforming Movement Paths in Wireless Mobile Network Simulation

Hawra Alseef, John DeDourek, Przemyslaw Pochec
Faculty of Computer Science
University of New Brunswick
Fredericton, Canada
e-mail: {hawra.alseef, dedourek, pochec}@unb.ca

***Abstract*—Mobility pattern of nodes significantly affects performance of MANETs. We propose a new investigative approach for mobile network performance evaluation based on modifying the mobile node movement path between the waypoints. We implemented the modifications to the movements generated using the standard *setdest* utility for the ns2 simulator, replacing line movements between the waypoints with the new node movements along the curved paths generated using simple fractals. Applying the new path modification approach in a study of a sample MANET showed that only the node speed significantly affected MANET performance, and not the shape of the individual path segments between the waypoints.**

Keywords-movement generator; network simulation; ns2; fractal path; MANET

I. INTRODUCTION

Mobility [1] is an important aspect of operation of any Mobile Ad hoc Network (MANET), where mobile devices cooperate with each other by exchanging messages and forwarding data [2][3]. Mobile devices are linked together through wireless connections without infrastructure and can change locations and reconfigure network connections. During the lifetime of the network, nodes are free to move around within the network and node mobility plays a very important role in mobile ad hoc network performance. Mobility of mobile nodes significantly affects the performance of a MANET [3].

Mobile Medium, or Mobile Medium Ad hoc Network (M2ANET) [4], is a particular type of mobile network that affords a significant flexibility in choosing a movement pattern for its nodes. M2ANET consists of two types of nodes: a small number of user nodes that actively send data one to another, and a large cloud of mobile nodes that act as relays and forward the data between the user nodes along multi hop paths. The mobile nodes act as a mobile infrastructure and can be set up to provide the best service for the user nodes. When setting up M2ANET, the administrator chooses the number of nodes required, the routing/forwarding protocol to be used by the Mobile Medium nodes and the mobile behavior of the forwarding nodes. The mobile nodes for M2ANET can be implemented with autonomous aerial drones carrying radio transceivers. M2ANETS are not yet common and experimenting with such a network would require controlling both physical objects in space (aerial drones) as well as data circulating in the network.

For preliminary evaluation of properties on the new M2ANETS network simulation can be used very effectively. Simulation allows for modelling existing networks as we as future networks. Using simulation, different mobile network configurations with mobile nodes following different movement patterns, working under different traffic load conditions and using different routing protocols, can be quickly and easily modelled and evaluated. For mobile networks, if no other design constraints are present, random motion of the nodes is usually used. Using a common and specific random motion model allows to create the base condition for comparison between different network evaluations.

ns2 is an open source simulator well suited for modelling wired and wireless networks [5]. It includes a motion scenario generator *setdest* designed to automatically generate random motion paths for a large number of nodes. This tool generates a random motion path for each node by selecting a random destination for the node and then moving the node towards this destination along a straight line. Once this destination is reached by the node, a new destination is randomly selected and, after an optional pause time, the node starts moving again to the new destination.

In this paper, we propose to use the random motion generated by the ns2 *setdest* utility [5] to create a new trajectory for the mobile nodes. The waypoints are kept the same but the path followed by the node between two waypoints is no longer defined by one straight line segment, and is replaced by a fractal curve composed of a number of shorter line segments. The manipulation of the node movement path for mobile networks has never been reported before. Incorporating this approach into mobile network simulation would allow to experiment with performance of similar mobile networks, checking how the performance would be affected by modifying the trajectories of mobile nodes, while keeping the waypoints the same. (The scenario is analogous to changing the flight paths of the aeroplanes while keeping the departure and destination cities the same.) The proposed approach, which is based on application of fractals, is particularly suitable for implementation in a discrete event simulator like ns2.

In Section II, we review different random movement models commonly used in simulation. Section III introduces properties of the Koch fractal. Section IV describes the new movement generator based on generation of the node movement along a fractal path. In Section V we introduce our simulation environment: the ns2 simulator. Section VI presents a study of two mobile networks: one with the conventional random movement and the other with the fractal movement. Conclusion is presented in Section VII.

II. STATE OF THE ART

Any model of a MANET requires a mobility model specifying the movement pattern of the nodes [6]. The most realistic models are trace driven but cannot be always applied because of their *a posteriori* nature. On the other hand, the synthetic models [6][7] are not trace driven but instead rely on assumptions about the node movement mode. The synthetic models attempt to generate the next node movement based on a goal driven scenario (e.g., reaching a particular destination) and some physical constrains on the movement of the node. The constraints may be presented in terms of geographical restrictions. The most common example of these is the movement area defined in the simulator typically as a rectangular region with the nodes not able to breach the boundary. This restriction is particularly important to consider when nodes are designed to follow curved path between waypoints; the simulation must assure that the curved path falls within the simulated region. In more complex scenarios the node movement planning would also have to avoid obstacles. Another type of movement restriction is a temporal dependency: the next move of a node is affected by the past movement. Typically, this restriction incorporates the constraints from the real world object dynamics, where physical systems cannot accelerate, or change direction, at an arbitrary rate, and allows for the implementation of more realistic simulation scenarios. The synthetic models may also incorporate special dependencies: the movement pf a node may depend on the movement of the other nodes around it. Two obvious scenarios could be considered: collision avoidance and group mobility. Group mobility is particularly attractive for MANETS where, in order for the nodes to maintain the communication links with each other, the nodes need to stay within the communication range of each other's' transmitter.

The synthetic model most often used mobile network proof concept simulation is the random mobility model where the nodes move randomly and without restrictions and where the destination and the speed are also chosen randomly. A random mobility model typically implements a rudimentary geographical restriction and keeps all the modes within a designated simulation region, typically a rectangle.

There are many different types of random mobility models that are used in MANETs. The main ones are the Random Walk, the Random Waypoint, and the Random Direction. The Random Walk model [7] mimics the Brownian motion of particles found in nature. Each node travels in a straight direction for a specified time interval before randomly changing the speed and the direction, and then continuing for another time interval. In the Random Waypoint model [8], each node selects a destination within the simulation area and then follows a straight path to it; once the destination is reached the node may pause and then select a new destination (waypoint). In the Random Direction model [9], instead of selecting a random destination, the node selects a random direction and then moves along this direction until it reaches the simulation area boundary where, possibly after a pause, it selects a new direction for the next move.

The ns2 *setdest* utility generates the node movements following the Random Waypoint algorithm [5]. In the Random Waypoint Movement (RWP), each node moves from its randomly selected initial starting position towards the randomly selected at a randomly selected speed. Once at the target destination the node may pause for a randomly selected time, and then start the next random move. This process will be repeated until the end of the simulation by the ns2. One notable aspect of the RWP movement is that the nodes following this pattern tend to concentrate in the center region of the deployment area [10][11].

While most studies use straight line piecewise motion for modelling mobile network node movements, investigating curved motion trajectories in mobile networks was a subject of a very few studies. Wang et al. [12] investigates the impact of the shape of the movement path on the efficiency of intrusion detection in a (battlefield) sensor network. An intruder can invade the network following a curved path, or even a random walk, in order to improve its network attacking probability. Wang's research describes the effects of different paths taken by the intruder, on the intrusion detection probability in an arbitrary wireless sensor network. Wang's study of the performance of a wireless mobile network would be the kind of investigation that could potentially benefit from the new approach described in our paper for transforming the existing movement paths into new ones.

III. THE KOCH FRACTAL

Fractal objects were first mentioned in 17th century and referred to as "fractional exponents" by Gottfried Leibniz when he explored the concepts of recursion and self-similarity [13]. In 1872, Karl Weierstrass proposed the definition of a curve, based on a function defined on the sum of Fourier series, that is everywhere continuous but nowhere differentiable, and could be characterised as a fractal. In 1883, Georg Cantor introduced the Cantor sets, which are examples of subsets of the real line that have unusual properties and are also considered fractals. In the last part of that century, Felix Klein and Henri Poincare discovered a number of fractal patterns that known as "self-inverse" fractals. In 1904, Helge Von Koch, introduced the

famous fractal, the Von Koch curve. The actual term "fractal' was introduced only in 1975 by B. Mandelbrot [14].

Fractals are complex patterns exhibiting self-similarity at different scales. Two most common types of fractals are complex number fractals and iterative function system (ITF) fractals. Mandelbrot and Julia sets are examples of fractals that are generated by iterating a recursive complex number formula. Koch snowflake and Sierpinski triangle are the examples of iterative function system fractals. The ITF fractals, when constructed in two dimensions, are of particular interest here, as they involve a transformation of an initial pattern on a plane. The initial pattern can be anything: a point, a line, a triangle etc., as well as the path taken by a mobile node in a wireless communication system.

We propose to use fractals for the movement generation for mobile network simulation based on the RWP model. Instead of moving the nodes along a straight line between the waypoints, the nodes are moved along a fractal path. We selected the Koch fractal because, like a line segment, it has a defined starting and ending points, and because of the simplicity of its generating algorithm [15][16].

A. *Construction of the Koch curve*

The construction of the Koch starts with a straight line that is then converted to the Koch fractal curve, Figure 1.

Figure 1. Step 1.

Figure 2. Step 2.

Figure 3. Step 3.

This process is then repeated for each of the 4 segments generated at the first iteration, leading to the curve shown in Figure 3. These steps can be applied repeatedly and eventually result in a complex shape. When the Koch curve generating algorithm is applied to an equilateral triangle it results in a closed curve called the Koch snowflake [16].

B. *Properties of the Koch snowflake*

Number of Sides (n): for each iteration, every segment of the curve from the previous iteration will be converted to four segments in the following iteration. Since we begin with three sides, the formula for the number of sides in the Koch curve is:

$$n = 3 * 4^a \quad (1)$$

where a indicates the number of iterations. For iterations 0, 1, 2, and 3, the numbers of sides are 3, 12, 48, and 192 respectively.

Length of Sides (L): In every iteration, the length of a side is 1/3 the length of a side from the previous iteration. If we begin with an equilateral triangle with side length x, then the length of a side in iteration a is:

$$L = x * 3^{-a} \quad (2)$$

For iterations 0 to 3, length = x, x/3, x/9, and x/27.

Perimeter (p): The key features of the Koch curve lies in having the same length of all sides in each iteration, this leads to a perimeter, which is simply the number of sides multiplied by the length of a side:

$$p = n * L \quad (3)$$

For the snowflake, from the previous formulas, we get:

$$p = (3 * 4^a) * (x * 3^{-a}) \quad (4)$$

In the same manner, for the first 4 iterations (0 to 3) the perimeter is 3x, 4x, 16x/3, and 64x/9. We notice that, the perimeter increases by 4/3 times for each iteration, so we can rewrite the formula as

$$p = (4/3)^a * 3x \quad (5)$$

IV. CUSTOM MOVEMENT GENERATION WITH FRACTALS

The main objective of this research is to propose and implement a new method for movement generation in MANET simulation in ns2. Indeed, the standard way for movement generation is to use the setdest utility that generates a set of setdest commands that are then "executed" in the ns2 simulator. setdest commands generate a movement along a straight line between the current location and the designated destination point. This research aims at providing a new tool for modifying the simulation environment by modeling motion in wireless network simulations, specifically for generating movement files for ns2 simulation that specify the motion along curved (fractal) paths. Typically, defining the node movements needs to be done ahead of the ns2 simulation. In general a curved path can be approximated by a series of short line segments, which determine the final shape of the curve. Therefore, a Java program was 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 (fractal). Once the new movement file is generated the ns2 simulation can proceed in a standard way.

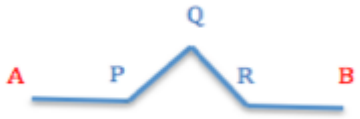

Figure 4. The result of fractal transformation of a line segment AB.

Let us consider the original setdest command for the direct movement from A to B (Figure 4):

$ns_ at T "$node_(#) setdest X_B Y_B S"

where T indicates the starting time at which the node starts moving towards the destination X_B,Y_B at the specified speed S. While splitting the initial path (line segment AB) into four segments (AP, PQ, QR and RB) and defining the destination of each of the four moves is a simple geometry, the other setdest command parameters require careful consideration. More precisely, the need of updating the time and speed in the setdest commands arises when applying the fractal transformation. In order to make the fractal movements arrive at the final destination (point B) at the same time that the original straight movement would have arrived, we need to do the following modifications:

$ns_ at T_P "$node_(#) setdest X_P Y_P S_{new} "
$ns_ at T_Q "$node_(#) setdest X_Q Y_Q S_{new} "
$ns_ at T_R "$node_(#) setdest X_R Y_R S_{new} "
$ns_ at T_B "$node_(#) setdest X_B Y_B S_{new} "

The four (fractal) movements should proceed sequentially, each having a starting time after the previous movement ends. To calculate the precise time of each move and the new speed we need to determine the new speed and the new starting time for each of the four new setdest commands. First, we need to calculate the time the node would take to travel from A to B at speed S along the original straight line path AB:

$$t_{AB} = \mathrm{sqrt}((X_B - X_A)^2+(Y_B-Y_A)^2)/S \quad (6)$$

then the start times for each move are calculated as:

$$\begin{aligned} T_P &= T \\ T_Q &= T_P + t_{AB}/4 \\ T_R &= T_Q + t_{AB}/4 \\ T_B &= T_R + t_{AB}/4 \end{aligned} \quad (7)$$

and the new speed, due to the distance travelled increased by 1/3, is:

$$S_{new} = 4 * S / 3 \quad (8)$$

For example, let us consider the following setdest statement taken from a movement file generated by the setdest utility:

$ns_ at 0.0 "$node_(1) setdest 900.0 0.0 10.0"

This line specifies that at time 0.0 s, node 1 starts to move from the starting point (0,0) towards the destination (900,0) at a speed of 10.0 m/s (this is one single movement in a straight line). When our fractal transformation is applied once to the movement path, 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 Koch fractal (i.e., one iteration of the Koch fractal generation algorithm). The four movements are listed below:

$ns_ at 0.0 "$node_(0) setdest 300.0 0.0 13.3"
$ns_ at 22.5 "$node_(0) setdest 450.0 260.0 13.3"
$ns_ at 45.0 "$node_(0) setdest 600.0 0.0 13.3"
$ns_ at 67.5 "$node_(0) setdest 900.0 0.0 13.3"

Applying the fractal transformation to these four new movements for the second time, results in 16 new movements:

$ns_ at 0.0 "$node_(0) setdest 100.0 0.0 15.029"
$ns_ at 5.625 "$node_(0) setdest 150.0 87.0 15.029"
$ns_ at 11.25 "$node_(0) setdest 200.0 0.0 15.029"
$ns_ at 16.875 "$node_(0) setdest 300.0 0.0 15.029"

$ns_ at 22.5 "$node_(0) setdest 350.0 87.0 15.029"
$ns_ at 28.125 "$node_(0) setdest 300.0 173.0 15.029"
$ns_ at 33.75 "$node_(0) setdest 400.0 173.0 15.029"
$ns_ at 39.375 "$node_(0) setdest 450.0 260.0 15.029"

$ns_ at 45.0 "$node_(0) setdest 500.0 173.0 15.029"
$ns_ at 50.625 "$node_(0) setdest 600.0 173.0 15.029"
$ns_ at 56.25 "$node_(0) setdest 550.0 87.0 15.029"
$ns_ at 61.875 "$node_(0) setdest 600.0 0.0 15.029"

$ns_ at 67.5 "$node_(0) setdest 700.0 0.0 15.029"
$ns_ at 73.125 "$node_(0) setdest 750.0 87.0 15.029"
$ns_ at 78.75 "$node_(0) setdest 800.0 0.0 15.029"
$ns_ at 84.375 "$node_(0) setdest 900.0 0.0 15.029"

Please note that, in the case of the Koch curve generation, standard geographical restrictions on the mobile node movements apply. When the calculated position of the intermediate point Q of the Koch fractal would fall outside the predefined simulation region (Figure 5), then the corresponding segment of the fractal is not generated, and the original straight line segment remains, Figure 6.

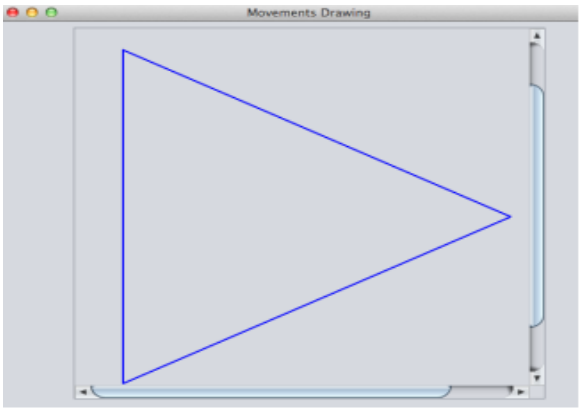

Figure 5. Trace of sample simple node movement.

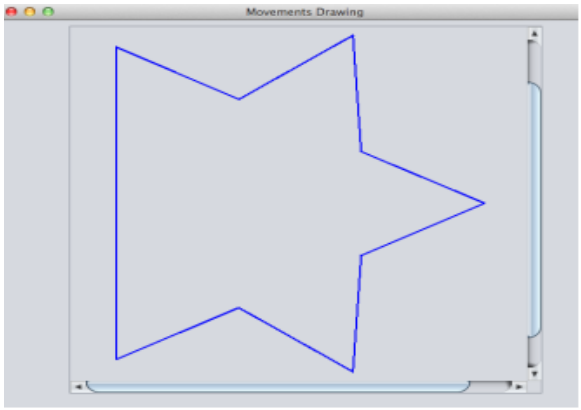

Figure 6. Fractal movement generated from Figure 5.

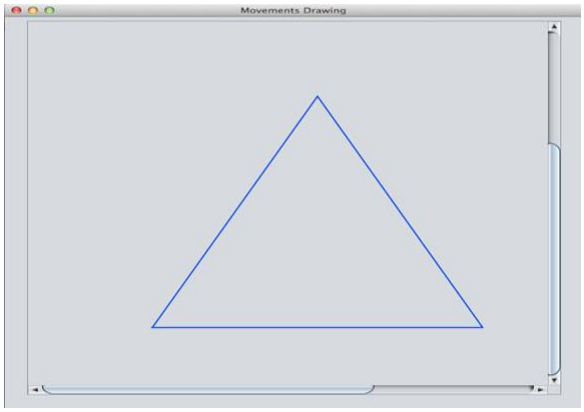

Figure 7. Sample original movement along the edges of a triangle.

In a more complex scenario, the effects of the new fractal path modification algorithm can be best illustrated graphically: when the transformation is applied three times to the movement file with three original movements defined along the sides of an equilateral triangle, the result is a curved movement defined with 192 straight line movements in the shape of the Koch snowflake (Figures 7 and 8).

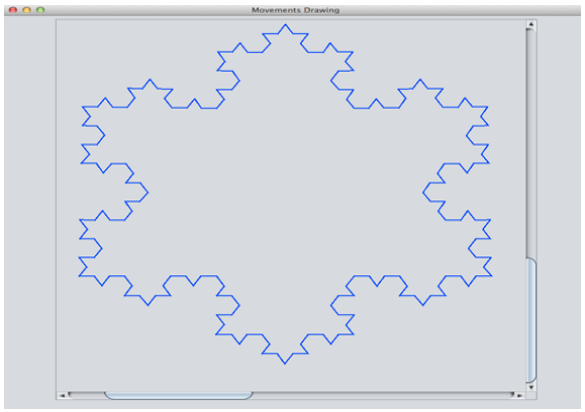

Figure 8. The same movement as in Figure 6 after three Koch fractal steps.

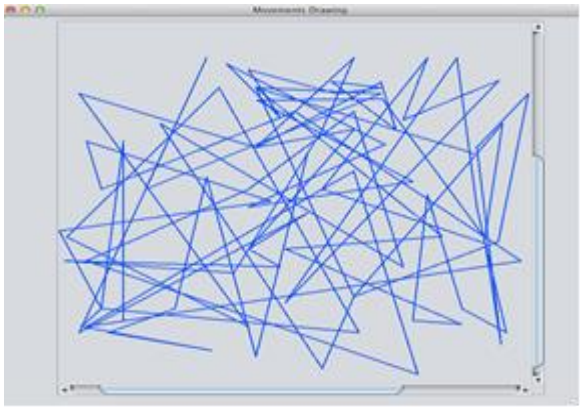

Figure 9. Trace of complex random movement.

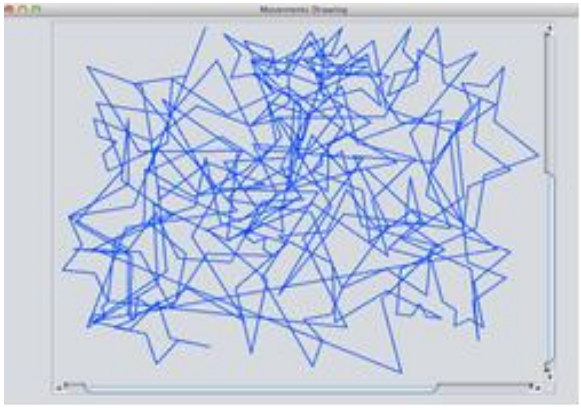

Figure 10. Fractal movement generated from Figure 8.

Finally, when the transformation is applied to a sample movement file generated with ns2 setdest utility, the original random movements, each along a straight line, Figure 9, are turned into fractal movements, four times as many, covering the same locations (waypoints) as in the original movement file (Figure 10).

V. EXPERIMENTAL ENVIRONMENT: NS2 SIMULATOR

The network simulator ns-2 is a popular open source simulation environment [4]. In a typical configuration, it consists of the simulator module ns2, the network animator module nam, the stand alone utility for generating wireless scenarios setdest and the plotting module xgraph. ns2 is an object oriented discrete event simulator written in C++ and OTcl. The event scheduler and the basic network component libraries are written in C++ for greater efficiency. OTcl is used for setting up and controlling the simulation. In a typical simulation session the user sets up the network nodes, links, transport protocols and traffic generators using an Otcl script and then runs the simulation. In case of simulating wireless mobile network, the user also specifies the node characteristics, the routing protocol and the node movements. To make it more convenient to define more complex wireless scenarios with a large number of nodes making many moves over an extended period of time (each move requires a separate command), ns2 installation includes the setdest tool. This tool automatically generates a wireless scenario, with a large number of individual node movements specified, based on a small set of user defined parameters. Setdest is particularly suitable for creating base case scenarios with a large number of nodes moving randomly over a rectangular regions and predefined range of speeds.

The simulation results are written into a trace files recording all the network events. The network animator tool nam displays the animation of the network's events from the trace file, showing the movement of nodes and packets in the network. The results can be plotted with xgraph as well. The trace files generated by ns2 are in the form of text files that can be viewed directly, or processed easily, to extract the desired network performance metrics.

ns2 includes a library of the most common transport and, in case of wireless mobile network simulation, routing protocols. Various versions of the Transposrt Control Protocol (TCP) and the User Datagram Protocol (UDP) are available for experimentation. For experimenting with the ad hoc mobile networks, the Destination Sequence Distance Vector (DSDV), Dynamic Source Routing (DSR), Temporally Ordered Routing Algorithm (TORA) and Ad hoc On-demand Distance Vector (AODV) are supported.

The AODV protocol [17][18] used in our experiments, is an ad hoc network reactive routing protocol. The node, wishing to transmit to another node, first broadcasts a route request (RREQ) message to the neighbouring nodes. This process continues until a RREQ arrives at the destination (or at a node that possesses a current route to the destination). As the RREQ traverses the network towards the destination, all the nodes in its path set up the reverse path back to the source. Once the destination is reached, a route reply packet (RREP) is sent back following the reverse path, signalling the establishment of the route to the source node. Under some conditions, AODV offers superior performance when compared to other established routing protocols [18].

VI. EVALUATION OF MANET PERFORMANCE UNDER FRACTAL MOVEMENT

We evaluated the performance of a sample MANET under different motion generation conditions. A MANET with the number of nodes ranging from 5 to 80 was simulated over the area of 800 by 800 meters. The MANET in the experiment was set up as the Mobile Medium Ad Hoc Network (M2ANET) [4]. Mobile Medium networks are a special case of MANETs, where the mobile network nodes are divide into two categories: the forwarding only nodes (shown in black in Figure 11) and the communicating nodes (shown in red) that use the rest of the Mobile Medium for multi hop communication. In our experiment, the M2ANET was set up with two stationary communicating stations located at (100,500) and (700, 500), Figure 11. Constant Bit Rate (CBR) traffic was generated over UDP and routed with the AODV protocol (Table I). Standard RWP movement was generated with setdest and then the standard movement was converted to the fractal movement using one step of Koch generating algorithm, as shown in Figures 9 and 10.

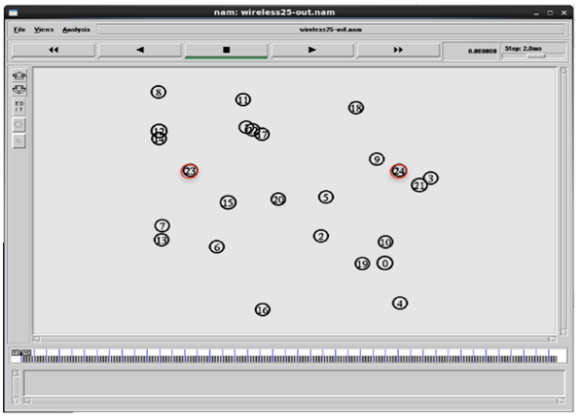

Figure 11. Screen shot of ns2 network animation in the nam utility, showing the mobile nodes and two fixed communication stations.

TABLE I. SIMULATION PARAMETERS

Parameters	
Simulator	NS-2.33
Channel Type	Channel / Wireless Channel
Network Interface Type	Phy/WirelessPhy
Mac Type	Mac/802.11
Radio-Propagation Type	Propagation/Two-ray ground
Interface Queue Type	Queue/Drop Tail
Link Layer Type	LL
Antenna	Antenna/Omni Antenna
Maximum Packet in ifq	50
Area (n * n)	800 x 800
Source Type	(UDP) CBR
Simulation Time	100s
Routing Protocol	AODV

Two scenarios were investigated: (i) low speed (10m/s), and (ii) high speed (30m/s).

Figure 12 illustrates the difference in the number of packets received at the destination when using the original movement and the new fractal movement at low speed. It shows that most of the time the packet delivery for the fractal movement is higher than the original linear movement. Although the speed of the fractal path is higher than the original (because of the increased path length along the fractal curve between the original waypoints), we observed a higher number of packets delivered at the destination for the fractal movement at speed of 13m/s. However, applying the t-test for the comparison of two paired means representing the packets received in the linear motion and the fractal motion experiments with 25 nodes gives 8%, which indicates that the observed difference is not statistically significant. Also, comparing the average packet delivery across all node densities does not show a significant difference (t-test value 49%).

Figure 13 shows the packet delivery for linear and fractal motions at high speed. This time we observe a lower packet delivery for fractal motion recorded in most of the experiments.

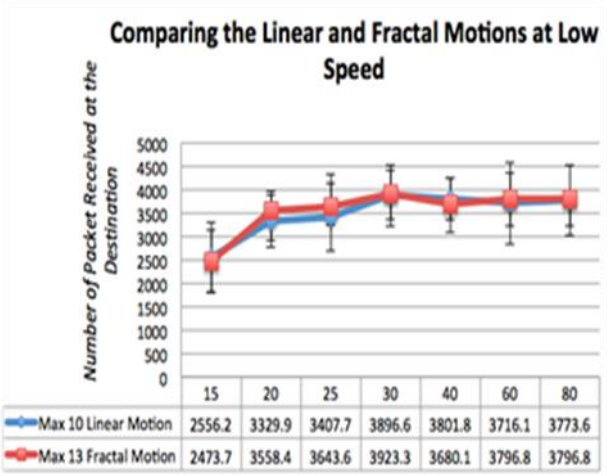

Figure 12. Throughput comparison at low speed

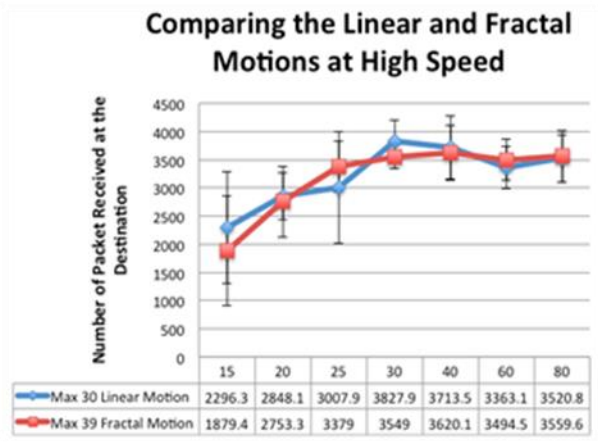

Figure 13. Throughput comparison at high speed.

One possible explanation of lower performance with fractal motion is that the increase in movement speed of 10m/s, from 30 to 40, results in more frequent link disconnections and consequently lower packet delivery. Applying the t-test for the comparison of two paired means representing the packets received in the linear motion and the fractal motion experiments with 25 nodes gives 32%, which indicates that the observed difference is not statistically significant. Also, comparing the average packet delivery across all node densities does not show a significant difference (t-test value 60%).

Figure 14 illustrates the advantage of using lower speed in a network with linear motion. It shows that the packet delivery is consistently higher at low speed for almost all node densities. Applying the t-test for the comparison of two means representing the packets received in the linear motion and the fractal motion experiments with 20 nodes gives 4%, which indicates that the observed difference is statistically significant. The average packet delivery for all node densities is 3176 at high speed and 3497 at low speed, and this difference in performance is statistically significant (t-test value 0.076%). This result is consistent with the performance of ad hoc routing protocols, like AODV, in a high mobility network [19]. At the network layer, packets are buffered and eventually dropped if the valid route to the destination node is not known at the forwarding node. At the MAC layer, the packets are dropped when the routing information is obsolete and the next hop node is out of range.

Similarly, Figure 15 illustrates the advantage of using lower speed in a network with the fractal motion. The packet delivery is consistently higher at low speed for all node densities. Applying the t-test for the comparison of two means representing the packets received in the linear motion and the fractal motion experiments with 20 nodes gives 1%, which indicates that the observed difference is statistically significant. The average packet delivery for all node densities is 3186 at high speed and 3553 at low speed, and this difference in performance is statistically significant (t-test value 1.7%).

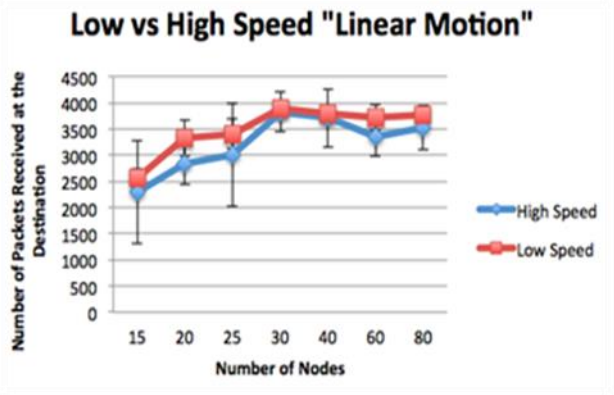

Figure 14. Throughput comparison for linear motion

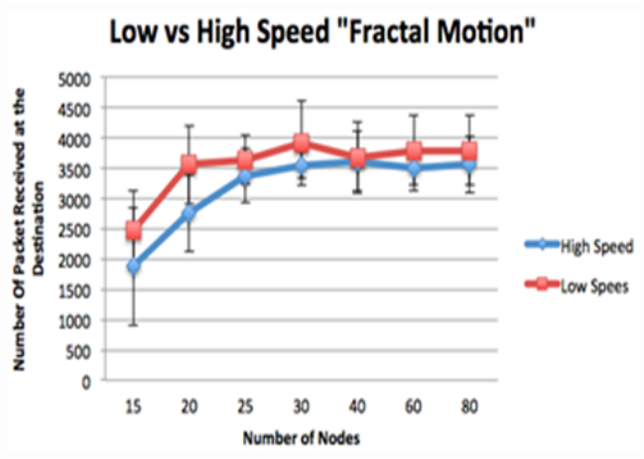

Figure 15. Throughput comparison for fractal motion.

VII. CONCLUSION AND FUTURE WORK

In this paper, we presented a new paradigm for investigating mobile wireless networks using simulation. While conventional simulation approaches focus on using different protocols, data generators and different movement patters, our new paradigm focusses on manipulation of the existing movement files. The existing movement files can either represent the actual node movements in a real network (e.g., trace based) or be automatically generated (e.g., random models). The new investigative paradigm aims at examining the changes in the network performance resulting from replacing the existing node movement pattern with a new one, that is obtained as a result of a transformation of the old movement pattern.

We presented a tool for transforming linear movements into fractal movements based on the Koch curve. The new tool reads a standard ns2 movement file, decodes each movement, and replaces it with a series of new movements forming a fractal curve, and then outputs a new movement file. The newly generated movement file satisfies the ns2 specifications and can be used in the ns2 simulator. Both, standard movement files generated with *setdest* and new movement files generated with the new fractal tool were used in simulating a MANET with varying number of nodes (i.e., with different node densities). The fractal transformation can be applied to any ns2 movement files, including the ones automatically generated with the *setdest* utility.

We used the new fractal transformation in ns2 simulation and compared the MANET performance in terms of packet delivery under two different motion scenarios and at different speeds. In the experiments, we observed marginally higher performance of MANET with fractal motion at low movement speeds, suggesting that using a curved path between the waypoints (rather than straight line) would offer an advantage in terms of network performance. However, the statistical tests show that the difference observed in our limited experiments is not significant. We also observed that the packet delivery is lower at higher speeds for both motion types, and after the application of the t-test for the difference of the means, we concluded that the observed lower packet delivery at higher speed is statistically significant. From our results, we conclude that, in the scenarios investigated, only the node speed significantly affects the MANET performance, and not the shape of the path taken by a node.

The work presented in this paper successfully demonstrated the new experimental approach for investigating performance of mobile networks, i.e., applying transformations to the node movement paths. The future work on transforming the node movement paths might include using more than on iteration of the generating function of the Koch fractal, calibrating the node speed when it starts moving on the new curved path and testing if the new path generators reduce the tendency observed in the RWP model of clustering the nodes towards the center of the experimental area. In general, the proposed new paradigm for mobile network simulation, that involves transformation of mobile nodes paths, could be applied in a study of mobile network intrusion detection, similar to Wang et al. [11]. In such a simulation study, we could specify only the destination of the rogue node penetrating a mobile network, and then use the approach similar to the one described in our paper, to generate multiple different paths to this destination, and then investigate the performance of intrusion detection under these different, and automatically generated, experimental scenarios.

ACKNOWLEDGMENT

This work is sponsored and funded by the Ministry of Higher Education of Saudi Arabia through the Saudi Arabian Cultural Bureau in Canada.

REFERENCES

[1] A. Alseef, J. DeDourek and P. Pochec, "A Method for Custom Movement Generation in Wireless Mobile Network Simulation", The Seventh International Conference on Emerging Networks and Systems Intelligence EMERGING 2015, Nice, France, July 19, 2015, pp. 27-32.

[2] S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic (Eds.), Mobile Ad Hoc Networking, New York, Wiley-IEEE Press, 2001.

[3] F. Bei and A. Helmy, A survey of mobility models in wireless Ad hoc Networks, University of California, USA, 2004.

[4] J. DeDourek and P. Pochec, "M2ANET: a Mobile Medium Ad Hoc Network", Wireless Sensor Networks: Theory and Practice, WSN 2011, Paris, France, Feb. 2011, pp. 1-4.

[5] H. Ekram and T. Issariyakul, Introduction to Network Simulator NS2, Springer, 2009.

[6] N. Aschenbruck, E. G. Padilla, and P. Martini, "A survey on mobility models for performance analysis in tactical mobile networks", Journal of Telecommunications and Information Technology, vol. 2, 2008, pp. 54-61.

[7] T. Camp, J. Boleng, and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research", Wireless Communication and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications , vol. 2, no. 5, 2002, pp. 483-502.

[8] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols", Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom98), ACM, October 1998, pp. 85-97.

[9] E. M. Royer, P. M. Melliar-Smith, and L. E. Moser, "An Analysis of the Optimum Node Density for Ad hoc Mobile Networks", Proceedings of the IEEE International Conference on Communications(ICC), Helsinki, Finland, June 2001, pp. 857-861.

[10] C. Bettstetter, "Mobility Modeling in Wireless Networks: Categorization, Smooth Movement, and Border Effects", ACM Mobile Computing and Communications Review, vol. 5, no. 3, July 2001, pp. 55-67.

[11] R. Alghamdi, Movement Generator for Mobile Network Simulation. Master's Report, Faculty of Computer Science, University of New Brunswick, Fredericton, Canada, 2012.

[12] Y. Wang, Y. K. Leow, and J. Yin, "Is Straight-line Path Always the Best for Intrusion Detection in Wireless Sensor Networks?", ICPADS '09, Proceedings of the 2009 15th International Conference on Parallel and Distributed Systems, IEEE Computer Society Washington, DC, USA, Dec. 2009, pp.564-571.

[13] B. B. Mandelbrot, The Fractal Geometry of Nature, Freeman 1982.

[14] B. B. Mandelbrot, Les Objets Fractals: Forme, Hasard et Dimension, Flammarion, Paris, 1975.

[15] G. Edgar, Measure, Topology, and Fractal Geometry, 2nd Ed., Springer 2008

[16] Koch's Snowflake, online, http://en.wikipedia.org/wiki /Koch_snowflake, retrieved: May 2015.

[17] C. Perkins, E. Belding-Royer, and D. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing", IETF, RFC 3561, https://tools.ietf.org/html/rfc3561, retrieved 2016-04-28.

[18] A. Tuteja, R. Gujral, and S. Thalia, "Comparative Performance Analysis of DSDV, AODV and DSR Routing Protocols in MANET Using NS2", ACE 2010, International Conference on Advances in Computer Engineering, 2010, pp. 330-333.

[19] Q. Zhao and H. Zhu, "An Optimized AODV Protocol in Mobile Ad Hoc Network", 4th International Conference on Wireless Communications, Networking and Mobile Computing, Dalian, China, Oct. 2008, pp. 1-4.