2011

Ns2 SetDest Acceleration Enhancement

EEC 787 Final Project – Mid Report

Brian Krupp and Limei Hou

Ns2 SetDest Acceleration Enhancement

Abstract

The introduction of the NS2 simulator has allowed students and researches to experiment existing and modifications of network protocols to help support their understanding and research. With the increasing presence of wireless networks and mobile nodes communicating within these networks, the simulation of these nodes is of greater focus. NS2 supports the capability of movement of mobile nodes via setdest commands in the simulator however these don't represent ideal movement scenarios, for example acceleration. Our goal is to implement this capability such that simulation models provide a more true representation of movement of a mobile node.

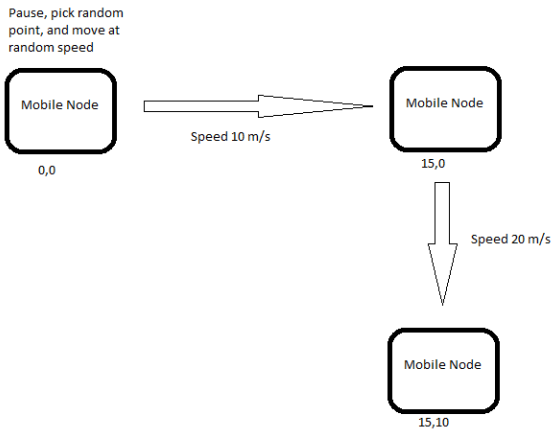
Introduction and Background

The NS2 simulator is a very powerful network simulation program that allows us to test network simulation of both wired and wireless networks. With the increasing presence of wireless network devices, often these devices are not stationary and we need to have a mechanism to test movement of these mobile nodes. The NS2 simulator provides the simulation capability of having an object move at a certain time to a specific coordinate from its current location at a specific constant speed. For example, the following code would move anode named WT(1) once the simulation reached five seconds to the coordinate 50,50 at a speed of 10 meters per second:

```
$ns_ at 5.0 "$WT(1) setdest 50.0 50.0 10.0"
```

The important piece of the code above being the setdest command. This code doesn't have any awareness of where the node is currently located at, only where it is going.

By using setdest, we are able to simulate movement within the ns2 simulator, but the entering the simulated movements for a set of particular nodes can become quite a task, especially if you have to do it for a simulation that contains a large set of nodes. Also if these commands were to be manually entered into a simulation script or sourced in, although the results would be consistent the movement is not random and may not provide a true simulation. To help address this problem, a Random Waypoint (RWP) mobility model was introduced into NS2. Random Waypoint mobility is the most popular in simulation studies, where each mobile node will follow the process of pausing at a location, selecting a waypoint randomly, and then move towards it with a randomly chosen speed between two numbers, then repeat this cycle until the end of simulation. The diagram below demonstrates this where the node starts off at coordinate 0,0, picks a random destination of 15,0 and moves there at a speed of 10 meters per second. The next coordinate it picks is 15,10, and picks a speed of 20 meters per second, and this cycle would then repeat until the end of the simulation.



To implement the above model in a simulation, two commands are needed for both movements. If you had twenty nodes, with each node moving five times during the simulation, you would have one hundred commands to put in the simulation.

To address this problem, a utility called “setdest” was developed in c++. The utility has two versions. The first version had an issue where when it picked a random speed, it would be so small that a node may never reach its destination and not provide a true simulation. Setdest version 2 fixed this problem by providing the capability of adding a minimum speed along with some other features. Below is an example of a setdest version 2 command that sets the version to 2, the nodes as 6, the speed type as 1, the minimum speed as 1, the maximum speed of 10, the maximum time of 200, pause type of 1, and the coordinates of our simulation graph. In the command below, we use a standard unix redirect to store the output from STDOUT to the mob file:

```
setdest -v2 -n 6 -s 1 -m 1 -M 10.0 -t 200 -P 1 -p 2.0 -x 1000 -y 1000 > mob
```

This mob file can them be sourced into a tcl script for NS simulation via the following commands:

```
set opt(sc) “./mob”
source $opt(sc)
```

The sourcing of the mob file should be after the nodes are defined. There is a problem within the setdest program that we found that needs to be modified in order to get this to work. GOD shouldn’t be declared as a variable in setdest, rather it should be referenced as a TCL class[1] so the below modifications had to be made:

Original:

```
#define GOD_FORMAT "$ns_ at %.12f \"$god_ set-dist %d %d %d\"\n"
#define GOD_FORMAT2 "$god_ set-dist %d %d %d\n"
```

Modified:

```
#define GOD_FORMAT "$ns_ at %.12f \"God set-dist %d %d %d\"\n"
#define GOD_FORMAT2 "God set-dist %d %d %d\n"
```

After these modifications were made, importing into the tcl script for NS2 simulation worked.

However, even with the convenience that the setdest utility brings, the movement is rigid with the mobile nodes. One of the features we will add is an acceleration feature that shows a node accelerating from its current location to its waypoint and decelerating as it reaches its destination. An example of where this feature can be useful is if you have wireless sensors attached to ground vehicles that move around and take measurements of their environment, this simulation would show the acceleration and deceleration of those vehicles and simulate real movement. There are many other examples where this can be used.

By default, NS2 doesn't inherently support acceleration and deceleration of nodes as they move with its sedest command. Our approach to implement this capability is to break a movement that is configured using setdest into several setdest commands so that we would get the behavior of as a node moves away from its origin, it does so at a slower speed, and once it hits a certain point in its path to its destination, that speed becomes constant, and as it approaches its destination, the speed decreases. For the increase and decrease, we will need to implement an approach where each setdest command that makes up each component slowly increases or decreases the speed.

Experiment Setup and Plan

To setup our experiment we will build from the lab 6 that we performed in our class. Our host machine will be an Ubuntu Virtual Machine running in VMware Player that has ns2 already installed on it and the source code for setdest. The host machine will be a Windows 7 machine.

We will first observe the simulations without our modifications, and then start to implement our acceleration into the mobile node. The first step will be to manually input the commands into the mobility file for a given node before it is automated into the setdest utility. We will then run the network simulation and see if the node that we modified moves as expected. For this experiment, we will use a low number of network nodes. If we are pleased with the movement, we will then begin to automate it within setdest by creating an acceleration and deceleration formula. Subsequent tests will check the movement of the node and then tune the algorithm so that the movement is fluid and more realistic.

Once we have the algorithm set, we will then parameterize the options into the algorithm by adding addition capabilities into setdest and allow a version 3. This will give the acceleration capability and other enhancements we may want to add in the future.

Expected Results and Comments

We expect that we can get acceleration and deceleration to work, however the primary concern is if the movement will appear to not be fluid in simulation since we are defining multiple setdest commands that may interfere with the fluidity of the movement from the origin to the destination. If this is something that other researches and students will find useful in ns2, the next step would be to allow it to be a core component within the ns2 where a setdest command would be modified so that an acceleration component can be modified into the simulation.

We also expect of getting the right formula for acceleration and deceleration to be a challenge as well. Some things to consider are the distance length, the top speed of the node, and how long it takes to accelerate and decelerate from the origin and to the destination. With this, we understand that depending on future enhancements, an acceleration/deceleration model may need to be imported into the network simulation as our acceleration/deceleration model will be very broad and can't be applied to each scenario.

References

[1] [ns] Setdest problem with $god_ (NS-2 v2.26) http://mailman.isi.edu/pipermail/ns-users/2003-December/038047.html