Improving the Accuracy of Random Waypoint Simulations Through Steady-State Initialization*

William Navidi, Tracy Camp, and Nick Bauer,
Department of Math. and Computer Sciences
Colorado School of Mines
Golden, CO 80401
wnavidi, tcamp, and nbauer@mines.edu

***Abstract*— In simulations of mobile ad hoc networks, the probability distribution governing the movement of the nodes typically varies over time, and converges to a "steady-state" distribution, known in the probability literature as the *stationary distribution*. Some published simulation results ignore this initialization discrepancy. For those results that attempt to account for this discrepancy, the practice is to discard an initial sequence of observations from a simulation in the hope that the remaining values will closely represent the stationary distribution. This approach is not always reliable. If, however, the initial locations and speeds of the nodes are chosen from the stationary distribution, convergence is immediate and no data need be discarded.**

Many published simulation results of mobile ad hoc networks use the Random Waypoint Mobility Model (the RWM model). In this paper, we show how to implement a steady-state mobility model generator (*mobgen-ss*) for the RWM model. We then show, via simulation results, that one is able to construct more reliable simulations for mobile ad hoc networks with *mobgen-ss*. Our *mobgen-ss* code is available at `http://toilers.mines.edu`.

Keyword: Simulations

I. Introduction

Mobile ad hoc networks are often studied through simulation, and their performance can depend heavily on the mobility model that governs the movement of the nodes [5]. In most cases, the probability distributions of the initial locations and speeds of the nodes differs from the corresponding distributions at later points in the simulation. In fact, it is generally true that the probability distributions of both location and speed vary continuously over time, and converge to a "steady-state" distribution, known in the probability literature as the *stationary* distribution. At any given point in the simulation, the distribution of location and speed is a weighted average of the initial distribution and the stationary distribution, with the weight shifting from the initial distribution to the stationary distribution as the simulation progresses.

Because the distributions of location and speed vary as a simulation progresses, the performance of network protocols can vary as well. In particular, network performance early in a simulation may differ substantially from the performance later in the simulation [19]. Up to now, the primary method for dealing with this problem (when the problem is addressed at all) has been to discard an initial sequence of observations [6]. The hope is that the values observed for location and speed past this initial sequence will have been sampled approximately from the stationary distribution. This approach has two drawbacks. First, it is inefficient, since it requires the discarding of data. Second, and more importantly, it is difficult to know just how long a sequence one needs to discard in order to be safely near stationarity. In fact, we show that convergence can take more than 1000 seconds of simulation time if the minimum speed is low.

We focus our discussion on the Random Waypoint Mobility Model [4], [12] (the RWM model), since it is the most common mobility model used in ad hoc network simulations (e.g., [7], [9], [11], [14]). In this model, each node is assigned an initial location (x_0, y_0), a destination (x_1, y_1), and a speed S. The points (x_0, y_0) and (x_1, y_1) are chosen independently and uniformly on the region in which the nodes move. The speed is chosen uniformly on an interval (v_0, v_1), independently of both the initial location and destination. After reaching the destination, a new destination is chosen from the uniform distribution, and a new speed is chosen uniformly on

*This work supported in part by NSF Grant ANI-0208352. Research group's URL is *http://toilers.mines.edu*. Reference for this manuscript: Technical Report MCS-03-08, The Colorado School of Mines, June 2003.

(v_0, v_1), independently of all previous destinations and speeds. Nodes may pause upon reaching each destination, or they may immediately begin traveling to the next destination without pausing. If they pause, the pause times are chosen independently of speed and location.

Virtually all published simulation results that use the RWM model begin with the nodes placed uniformly in the simulation area. Of course, this initial random distribution of nodes is *not representative of the manner in which nodes distribute themselves when moving*. The stationary distributions of location and speed in the RWM model are in fact quite different from the uniform distribution. In particular it has been noticed [3], [17], [19] that the stationary distribution of the location of a node is more concentrated near the center of the region in which the nodes move, since nodes traveling between uniformly chosen points spend more time near the center than near the edges. Yoon et al. [19] noticed that the stationary distribution of the speed differs from the uniform as well, and showed in particular that if the minimum speed v_0 is taken to be 0, the mean node speed approaches 0. A variant of the RWM model is presented in [1], but the stationary distribution for this model differs from the uniform as well.

One implementation of the RWM model (*setdest* for NS2 [10]) begins with a pause at the initial location [4], [16]. In other words, once the initial locations are uniformly chosen, simulations that use *setdest* have nodes begin paused at their initial locations (the pause time is a constant). Another implementation of the RWM model (*mobgen* for NS2 [10]) begins with (approximately) half the nodes moving and half the nodes paused [6] (the pause time is chosen from a uniform distribution); thus, for (approximately) half the nodes, the first pause occurs upon reaching the first destination. For this reason, simulations using *setdest* take longer to converge than simulations using *mobgen* (see Section III).

In [5], the authors present three approaches to the initialization problem. The first is to save the locations of the nodes after a simulation has executed long enough to be past the period of high variability, and use this position file as the initial starting point of the nodes. By creating many such position files, and starting each simulation trial with a different one, each simulation trial is started from a distribution close to stationarity. The second approach, which is essentially equivalent to the first, is to discard an initial number of seconds of simulation time produced by the RWM model in each simulation trial. The authors suggest that discarding 1000 seconds of simulation time (regardless of the node's speed) will ensure that the initialization problem is removed. While this is true for many simulations, we show below that convergence can take more than 1000 seconds of simulation time if the minimum speed is low. This points out one of the difficulties with these two approaches; it is difficult to know just how long a sequence one needs to discard. The third approach proposed in [5] is to assign initial positions and speeds to the nodes according to the distribution they will come to have over time, i.e., the stationary distribution.

In [15], we derive the stationary distributions for speed, location, and pause time for a node moving in a rectangular area under the RWM model. We note that only the initial location and speed (and pause time, if applicable) need to be sampled from the stationary distribution; all subsequent node destinations, speeds and pause times should be sampled from the uniform distribution.

In this paper, we show how we've implemented a steady-state mobility model generator (*mobgen-ss*) using the stationary distributions for the RWM model developed in [15]. We then show, via simulation results, that one is able to construct more reliable simulations for mobile ad hoc networks with *mobgen-ss*. Our *mobgen-ss* program (which is available from `http://toilers.mines.edu`) can be used to generate mobility files for two different simulators (both NS2 [10] and QualNet [18]) and for the interactive plotting program gnuplot.

II. The Steady-State RWM Model

A. setdest and mobgen

When using *setdest*, without pausing, the user gives a minimum and maximum speed (s_0, s_1) for nodes moving in the simulation via the RWM model. The *setdest* program then chooses a value s uniformly on the interval $(0, s_1)$. If $s \geq s_0$, s becomes the speed. If $s < s_0$, s_0 is chosen as the speed. Thus, for setdest, the mean initial speed is

$$\mu_{0,setdest} = \frac{s_1^2 + s_0^2}{2 s_1},$$

the steady state speed distribution is

$$f_{setdest}(s) = \begin{cases} \dfrac{1}{\ln s_1 - \ln s_0 + 1} & s = s_0 \\ \dfrac{1}{s(\ln s_1 - \ln s_0 + 1)} & s_0 < s < s_1 \end{cases}$$

and the steady state average node speed is

$$\mu_{setdest} = \frac{s_1}{\ln s_1 - \ln s_0 + 1}.$$

When using *mobgen*, without pausing, the user also gives a minimum and maximum speed (s_0, s_1) for nodes moving in the simulation via the RWM model. The *mobgen* program then uniformly chooses a speed, s, on the interval (s_0, s_1). The mean initial speed is therefore

$$\mu_{0,mobgen} = \frac{s_1 + s_0}{2},$$

the steady state speed distribution is

$$f_{mobgen}(s) = \frac{1}{s(\ln s_1 - \ln s_0)}$$

and the steady state average node speed is

$$\mu_{mobgen} = \frac{s_1 - s_0}{\ln s_1 - \ln s_0}.$$

We note that, on average, nodes move more slowly under *setdest* than under *mobgen*. We also note that the average initial speed for both *setdest* and *mobgen* is considerably greater than their corresponding steady-state average. For this reason, the average speed decreases from the initial value to the steady-state average in all simulations that use *setdest* or *mobgen*. The time needed for convergence to the steady-state average depends largely on the minimum speed s_0 [19]: the smaller s_0, the more time is needed for convergence. In Section III, we show that more than 1000 seconds of simulation time may be needed to reach steady state [15].

With *setdest* and *mobgen*, the spatial distribution of the nodes changes over time as well. The initial distribution is uniform on the network, and converges to a distribution that is more concentrated in the center as time passes [2], [3], [15].

While node speeds and locations are converging to their steady-state distributions, values of performance metrics for a given protocol, which are influenced by the distribution of speed and location, are converging to steady-state values as well. For this reason, when using *setdest* or *mobgen*, network performance can vary systematically with time, and measures of performance gathered during the convergence period may not accurately reflect long-term values.

The convergence period characteristic of *setdest* and *mobgen* can be eliminated by choosing initial speeds and positions from their corresponding steady-state distribution rather than from the uniform distribution [15]. In this way, the speeds and locations have their steady-state distributions from the start of the simulation. We refer to this as the *steady-state* RWM model.

As mentioned, the *setdest* program chooses a speed, s, uniformly on the interval $(0, s_1)$. If $s \geq s_0$, then s becomes the speed. If $s < s_0$, then s_0 is chosen as the speed. Since the *mobgen* program uniformly chooses a speed on the interval (s_0, s_1), we derive the steady-state model for *mobgen* instead of for *setdest*.

B. mobgen-ss *Without Pausing*

As discussed in Section I, the primary method for dealing with the initialization problem of the RWM model (if the problem is addressed at all) has been to discard an initial sequence of observations. To avoid this inefficiency in discarding data, it is only necessary to sample the initial speed and location from their stationary distributions. Subsequent speeds and locations should then be sampled from the uniform distribution.

To choose the initial speed s, choose U uniformly on $(0,1)$, then let $s = F^{-1}(U)$ (where $F^{-1}(U)$ is the inverse for the cumulative distribution function of the stationary distribution of S), such that

$$F^{-1}(u) = \frac{s_1^u}{s_0^{u-1}}. \tag{1}$$

See [15] for details.

An initial location can be chosen in two simple steps: choose an initial path and then choose a point on that path uniformly. The probability density of any chosen path is proportional to the expected time spent on that path. Since the speed is independent of the path, the expected length of time spent on any given path is equal to the length of the path divided by the expected speed. Thus, the probability density of any path is proportional to its length. The initial path is therefore chosen by choosing endpoints (x_1, y_1), and (x_2, y_2) in such a way that the joint probability density of these two points is proportional to the distance between them.

For the sake of simplicity, we assume that the network region is the unit square. It is straightforward to adjust the scaling to apply our results to any rectangular network. For example, if the pause time is zero, the location of the node in the unit square is (x, y), and the simulation area is a rectangle of size 300 m $\times$ 600 m, then the node's location in the simulation area is $(300x, 600y)$. For a non-rectangular region, the stationary distribution of location will differ from that of a rectangular network, and must be derived separately.

A convenient way to choose an initial location is by rejection sampling. First choose two points (x_1, y_1) and (x_2, y_2) uniformly on the unit square. Compute the length of the path between these two points, and divide by the length of the longest possible path, which is $\sqrt{2}$. Call this quotient r. Generate a random uniform variable U on $(0,1)$. If $U < r$, then accept (x_1,y_1) and (x_2,y_2) as the endpoints of the initial path. Otherwise, start over again with new values of (x_1,y_1), and (x_2,y_2). Once the endpoints of the path have been determined, the initial location of the node is chosen at random uniformly from the points on the path.

The following seven steps give a step-by-step summary of our procedure:

1) Generate (x_1,y_1), and (x_2,y_2) uniformly on the unit square.
2) Compute $r = [(x_2-x_1)^2 + (y_2-y_1)^2]^{1/2}/\sqrt{2}$.
3) Generate a random value U_1 uniformly on $(0,\ 1)$.
4) If $U_1 < r$, then accept (x_1,y_1), and (x_2,y_2). Otherwise, go to step 1.
5) Generate a random value U_2 uniformly on $(0,\ 1)$.
6) The initial location for the node on a unit square, (x_0,y_0), is $(U_2x_1 + (1-U_2)x_2,\ U_2y_1 + (1-U_2)y_2)$. If $(width, height)$ is the simulation area size, then the initial location for the node on the simulation area is $(width * x_0, height * y_0)$.
7) The node then travels to $(width * x_2, height * y_2)$ at the initially chosen speed. Upon reaching $(width * x_2, height * y_2)$, subsequent speeds and destinations are chosen from the uniform distribution.

In our simulations, we rejected about 60% of the candidates for the initial path. In other words, we had to generate an average of 2.5 candidate paths for each node to get an acceptable initial path. Since rejection sampling is employed only for the initial path, and not for any subsequent path, the extra time added to the simulation was quite small.

C. mobgen-ss *With Pausing*

As discussed in Section I, simulations that use *setdest* have nodes begin in a paused state and simulations that use *mobgen* have (approximately) half the nodes begin in a moving state[1]. To simulate the stationary distribution of the RWM model with non-zero pause time, some nodes should begin in a paused state and other nodes should begin in a moving state. The percentage of nodes that begin in a paused (moving) state depends on the values of the simulation parameters. In other words, to begin a simulation at the stationary distribution with pausing, the first thing to do is to determine, for each node, whether the node will begin in a paused state or in a moving state.

To accomplish this goal, choose U uniformly on (0,1) for each node. If $U < P_{pause}$ the node will begin from a paused state; otherwise it begins in motion. (P_{pause} is the long-run proportion of time a node is paused. See [15] for details.) If the node begins in motion, the procedure for choosing the initial position and speed are the seven steps given in Section II-B for the implementation without pausing.

If the node is to begin from a paused state, it is necessary to determine the length of time, P_0, of its initial pause. To achieve stationarity, P_0 must be the length of time from a point chosen at random from within a pause period until the end of that pause period. Suppose $h(p)$ denotes the probability density function of the pause time P and $H(p)$ is the cumulative distribution function associated with $h(p)$. By a fundamental result in renewal theory [8], the cumulative distribution function of P_0 is

$$H_0(p) = \frac{\int_0^p [1-H(t)]\,dt}{E(P)}, \tag{2}$$

where $E(P)$ is the expected length of a pause.

To sample P_0 from the cumulative distribution function $H_0(p)$, it is necessary to compute the inverse H_0^{-1}. Then choose U uniformly on (0,1) and let $P_0 = H_0^{-1}(U)$. The initial position of a node that begins paused is chosen uniformly on the simulation area. The node then remains at this location for a length of time equal to P_0. The initial speed of the node, once the period P_0 is over, is chosen uniformly on (s_0,s_1).

As an example, suppose the pause time P is distributed uniformly on $(0,P_{max})$. The cumulative distribution function of P is then $H(t) = t/P_{max}$ for $0 < t < P_{max}$, and $E(P) = P_{max}/2$. Computing the integral in the above equation yields

$$H_0(p) = [2pP_{max} - p^2]/P_{max}^2. \tag{3}$$

Inverting yields

$$H_0^{-1}(u) = P_{max}(1 - \sqrt{1-u^2}). \tag{4}$$

Therefore to choose P_0, choose U uniformly on $(0,1)$ and let

$$P_0 = P_{max}(1 - \sqrt{1-U^2}). \tag{5}$$

[1]For this reason, simulations using *setdest* take longer to converge than simulations using *mobgen* (see Section III).

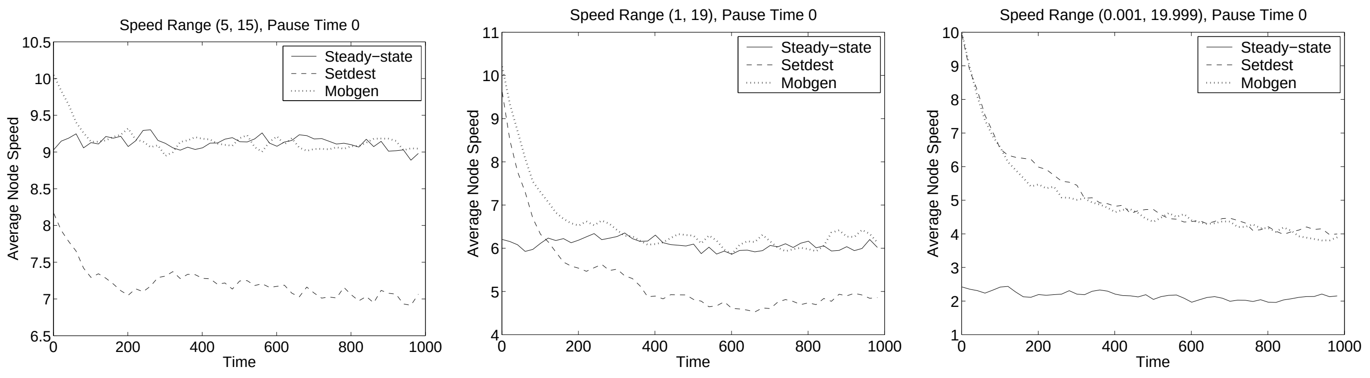

Fig. 1. Average node speed as a function of time for three mobility models.

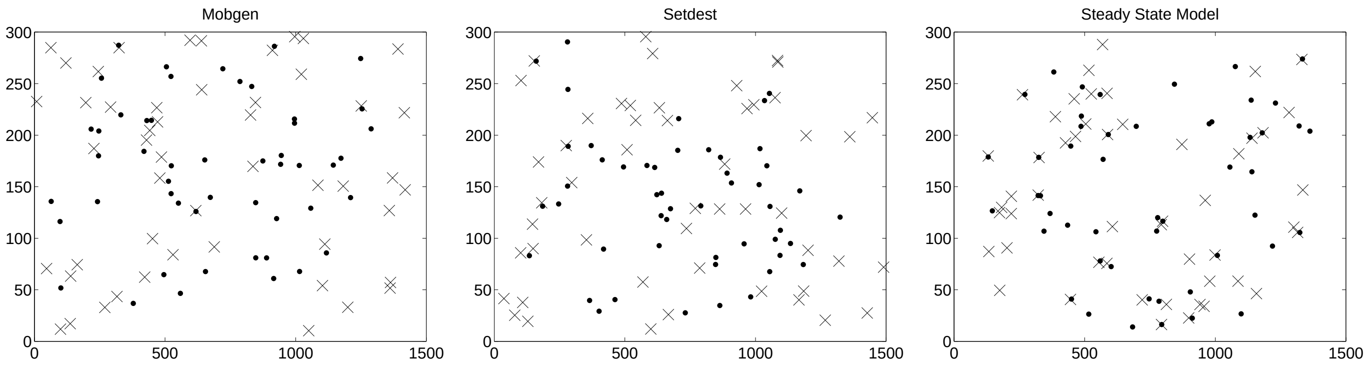

Fig. 2. Locations of 50 nodes both initially ($\times$) and after 500 seconds of movement ($\cdot$).

III. Simulation Results

A. *Convergence of* setdest, mobgen, *and* mobgen-ss

We demonstrate, via simulation results, that when the minimum speed is low, *setdest* and *mobgen* can produce unreliable performance metrics for a given protocol for more than 1000 seconds of simulation time. The steady-state model, on the other hand, produces metrics that reflect long term performance accurately after just a few seconds of simulation time.

1) Convergence of Average Speed: Figure 1 presents the average speed of 50 nodes in a 1500×300 network, plotted every 20 seconds for 980 seconds for *setdest*, *mobgen*, and *mobgen-ss* (listed on the figures as “Steady-state”). Speeds were chosen from three different ranges: (5, 15), (1, 19), and (0.001, 19.999), with no pause time. The figure shows that the average initial speeds for both *setdest* and *mobgen* agree well with their theoretical values. (See $\mu_{setdest}$ and μ_{mobgen} in Section II-A.) For example, for the speed range (5,15), the *setdest* average initial speed is 8.333 m/s and the *mobgen* average initial speed is 10 m/s.

For the speed ranges (5, 15) and (1, 19), the average speed for both *setdest* and *mobgen* converge to their steady-state values within 100-400 seconds of simulation time. For example, for the speed range (1,19), *setdest* converges to its steady-state average value (4.817 m/s) around simulation time 400 seconds and *mobgen* converges to its steady-state average value (6.113 m/s) around simulation time 300 seconds. For the speed range (0.001, 19.999), the convergence is not complete, even after 1000 seconds. (Our results are statistically significant at the 5% level.) We note that such a small minimum speed (i.e., 0.001) means that *setdest* and *mobgen* will generate (almost) equivalent mobility patterns. In fact, the steady state average node speed for the speed range (0.001, 19.999) is 1.834 m/s for *setdest* and 2.019 m/s for *mobgen*. Thus, when *setdest* and *mobgen* reach steady state for the speed range (0.001, 19.999), their average node speed should be (approximately) the average node speed shown for *mobgen-ss* (i.e., the “Steady-state” results in Figure 1).

For the steady-state model, the average speed is equivalent to the *mobgen* steady state average node speed (i.e., μ_{mobgen} given in Section II-A) throughout the simulation period for each of the speed ranges. As mentioned, the steady state average node speed for the speed range (0.001, 19.999) is 2.019 m/s.

2) Location Distribution: Figure 2 presents the locations of 50 nodes in a 1500×300 network, both after 1 second and after 500 seconds of movement, for *setdest*, *mobgen*, and *mobgen-ss*. The speed range was (0.001, 19.999) with no pause time. Similar results (not shown) would occur if we had done this for other speed ranges and pause times.

We note that the number of nodes near the edge of the network is much greater initially than after 500 seconds for both *setdest* and *mobgen*, reflecting the difference between the initial uniform distribution and the steady-state distribution that is eventually reached. (Convergence of both *setdest* and *mobgen* to steady-state locations is around 100 seconds [15].) When the initial locations are chosen from the steady-state distribution, however, the initial distribution does not differ in any systematic way from the distribution after 500 seconds of movement.

3) Convergence with Pausing: Sections III-A.1 and III-A.2 consider the convergence of *setdest*, *mobgen*, and *mobgen-ss* when pause time is equal to zero. If pause time is set to be greater than zero, then the average speed of *setdest* and *mobgen* is even slower. Thus, *setdest* and *mobgen* will take even longer to converge to their steady state values.

B. Routing Performance Comparison

We compare several performance characteristics of the steady-state model to those of *setdest* and *mobgen*. We show through simulations that when the initial speed s_0 is sufficiently small, the performance characteristics of *setdest* and *mobgen* vary systematically for more than 1000 seconds of simulation time. For larger values of s_0, the convergence time is shorter, and the performance characteristics of *setdest* and *mobgen* stabilize more quickly.

Our simulations involve a rectangular network with dimensions 1500×300, 50 nodes, and 30 communicating node pairs. Four 64-byte packets were sent by each communicating node each second. See Table I for a summary of our simulation parameters.

The routing protocol simulated was the Dynamic Source Routing (DSR) [12], [13]. DSR is a source routing protocol which determines routes on demand [12]. In a source routing protocol, each packet carries the full route (a sequenced list of nodes) that the packet should be able to traverse in its header. In an on demand routing protocol (or reactive protocol), a route to a destination is requested only when there is data to send to that destination and a route to that destination is unknown or expired. In the evaluation of DSR, both [4] and [11] only locate routes that consist of bi-directional links. The version of DSR in our study also only locates bi-directional links. In other words, a route reply packet containing the complete route from *S* to *D* is sent along the reverse route from *D* to *S*. A version of DSR from [16] was used for our simulations. The constants chosen for DSR's parameters are the same as those used in [4] and [11] (see Table II).

TABLE I
SIMULATION PARAMETERS

Simulator	NS2
Simulation time	1000s
Simulation area	1500m x 300m
Number of MNs	50
Transmission range	100m
Movement model	random waypoint
Speed	(1 m/s, 19 m/s) or (0.001 m/s, 19.999 m/s)
Pause time	0 or 30 s
CBR sources	30
Data payload	64 bytes
Packet rate	4 packets/s
Traffic pattern	peer-to-peer

TABLE II
DSR CONSTANTS

Timeout for 1 hop route request	30 ms
Retransmit route request	500 ms
Size of header with n addresses	4n + 4 bytes
Buffer size	64 packets
Packet lifetime in buffer	30 s

1) Steady-State Traffic Patterns Without Pausing: We first show that performance metrics will be unreliable for the first few simulation seconds, until traffic patterns stabilize, regardless of the mobility model used. Figure 3 presents the number of routing packets transmitted, the number of dropped data packets, and the average end-to-end delay each second for the first 50 seconds of simulation time. The speed range is (1, 19), with no pausing. The behavior shown is similar for all three mobility models. Routing packets transmitted and average end-to-end delay fluctuate greatly during the first few seconds as the initial packets are sent and routing tables

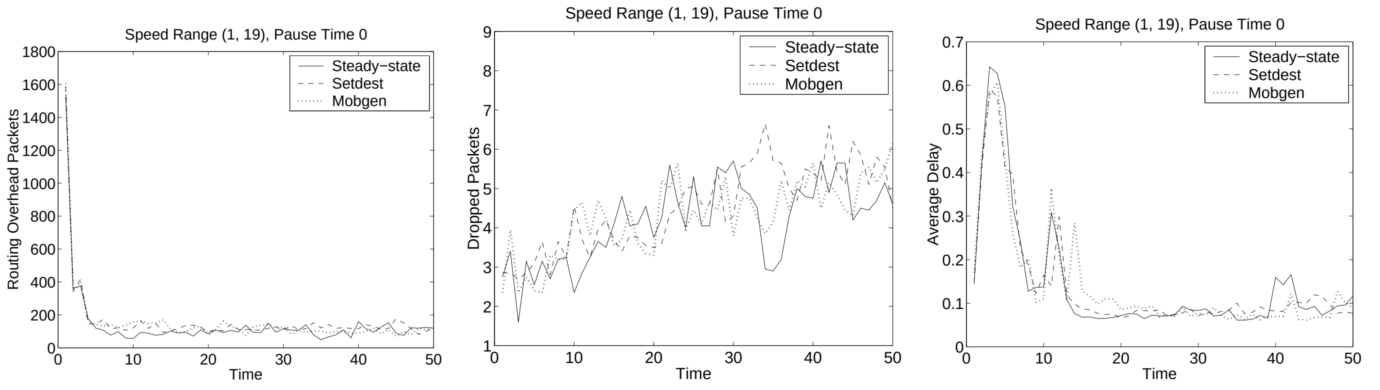

Fig. 3. Routing packets transmitted, dropped data packets, and average end-to-end delay for the first 50 seconds for three mobility models.

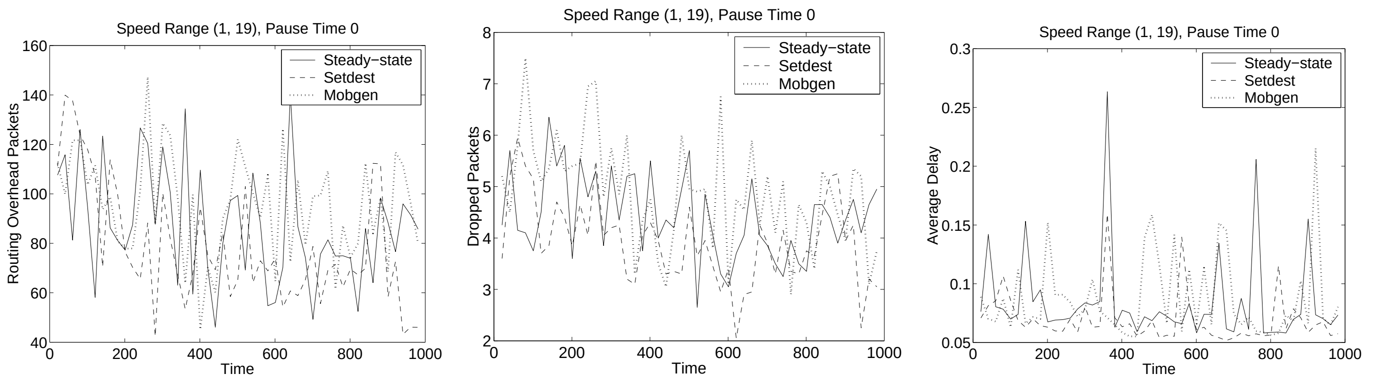

Fig. 4. Routing packets transmitted, dropped data packets, and average end-to-end delay for seconds 21 through 999 for three mobility models.

are constructed. The results for dropped data packets do not fluctuate as much, but appear to increase gradually for the first 50 seconds or so. Similar results (not shown) occurred for speed ranges (5, 15) and (0.001, 19.999) when there was no pausing; the one exception is that the number of dropped data packets did not increase for quite as long for the speed range (0.001, 19.999).

These early fluctuations are not related to mobility; they occur while the traffic pattern is stabilizing. However, if a simulation is executed long enough, the performance results of a given protocol will reach steady state; that is, these early fluctuations become insignificant if a simulation executes long enough.

2) Performance Comparison Without Pausing: Figure 4 presents the number of routing packets transmitted, the number of dropped data packets, and the average end-to-end delay each second for seconds 21 through 999 of the simulation for the speed range (1, 19) with no pausing. Values are plotted every 20 seconds. The performance is about the same for all three mobility models.

Figure 5 presents the number of routing packets transmitted, the number of dropped data packets, and the average end-to-end delay each second for seconds 21 through 999 of the simulation for the speed range (0.001, 19.999) with no pausing. Values are plotted every 20 seconds. Again, we note that such a small s_0 means the mobility patterns generated by *setdest* and *mobgen* are (basically) equivalent; thus, the performance of DSR should be equivalent to *mobgen-ss* (shown as "Steady-state" on the figures) if steady-state has been reached.

As shown in Figure 5, the performance for routing packets transmitted and dropped data packets is stable over time for the steady-state model, but varying systematically with time for *setdest* and *mobgen*, slowly converging to that of the steady-state model. Even after 1000 seconds, the convergence is not complete. (These results are statistically significant at the 5% level.) The performance of the steady-state model is generally better, because the average speed is slower. If the simulations were continued beyond 1000 seconds, the performances of *setdest* and *mobgen* would continue to improve, until they matched the values of the steady-state method. For

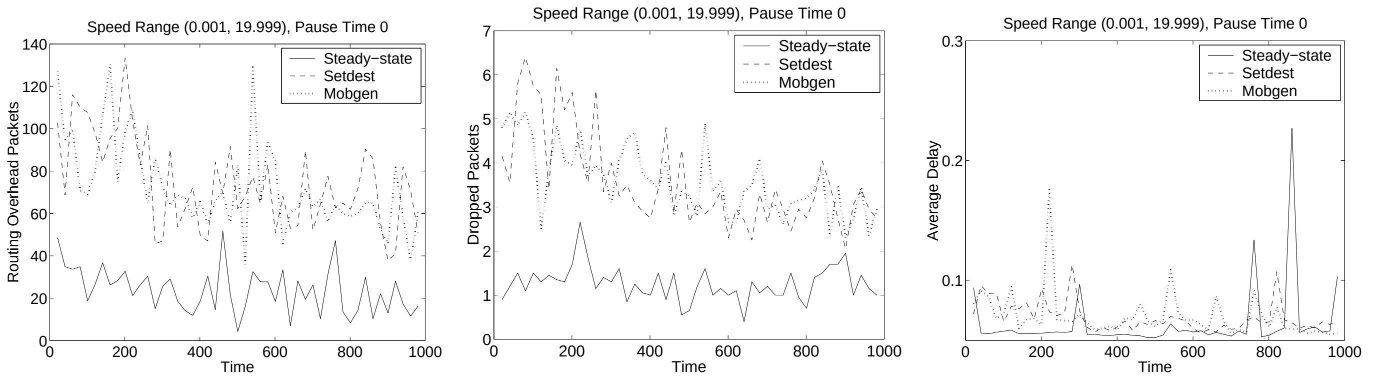

Fig. 5. Routing packets transmitted, dropped data packets, and average end-to-end delay for seconds 21 through 999 for three mobility models.

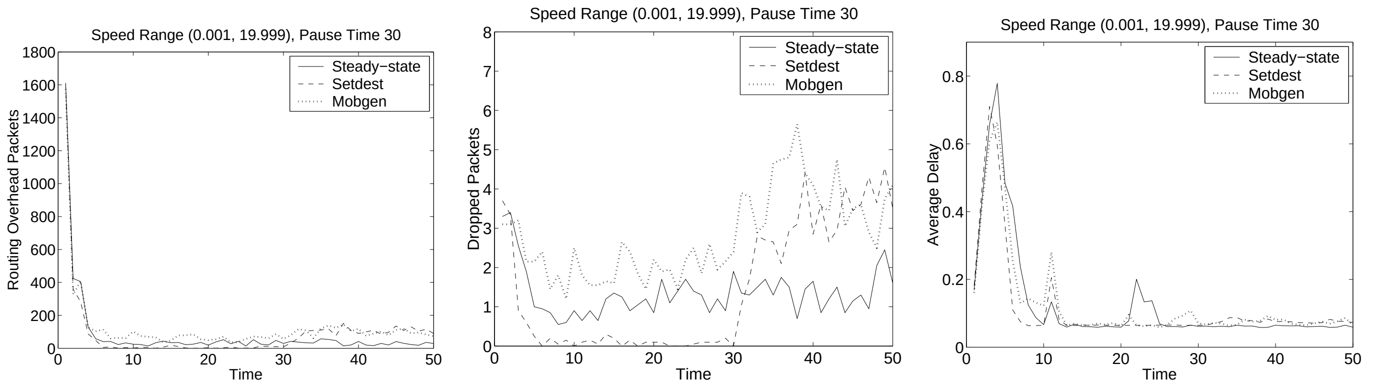

Fig. 6. Routing packets transmitted, dropped data packets, and average end-to-end delay for the first 50 seconds for three mobility models.

average end-to-end delay, there does not seem to be any time trends for any of the mobility models.

3) Steady-State Traffic Patterns With Pausing: Figure 6 presents the number of routing packets transmitted, the number of dropped data packets, and the average end-to-end delay each second for the first 50 seconds when the speed range is (0.001, 19.999) with a constant pause time of 30 seconds. The behavior during the first few seconds is similar to the no-pause scenario and reflects a brief period of traffic instability. The number of dropped data packets becomes very small for *setdest* after about 5 seconds, and remains small until the 30th second. This is due to the fact that the nodes begin in the paused state, and remain there for the duration of the 30-second pause time. In other words, for the first 30 seconds, *setdest* produces a static, rather than a mobile, network; thus the routing information learned during the first 5 seconds did not become obsolete until 30 seconds had passed in the simulation. After 30 seconds, the nodes begin moving, and the performance of *setdest* quickly reverts to that of *mobgen*.

4) Performance Comparison With Pausing: Figure 7 presents the number of routing packets transmitted, the number of dropped data packets, and the average end-to-end delay each second for seconds 21 through 999 for the speed range (0.001, 19.999) with a pause time of 30 seconds. Values are plotted every 20 seconds. Again, such a small s_0 means the mobility patterns generated by *setdest* and *mobgen* are (basically) equivalent; thus, the performance of DSR should be equivalent to *mobgen-ss* if steady-state has been reached.

For routing packets transmitted and dropped data packets, the performance is stable over time for the steady-state model, but varying systematically with time for *setdest* and *mobgen*, slowly converging to that of the steady-state model. Even after 1000 seconds, the convergence is not complete. (These results are statistically significant at the 5% level.) For average end-to-end delay, there does not seem to be any time trends for any of the mobility models.

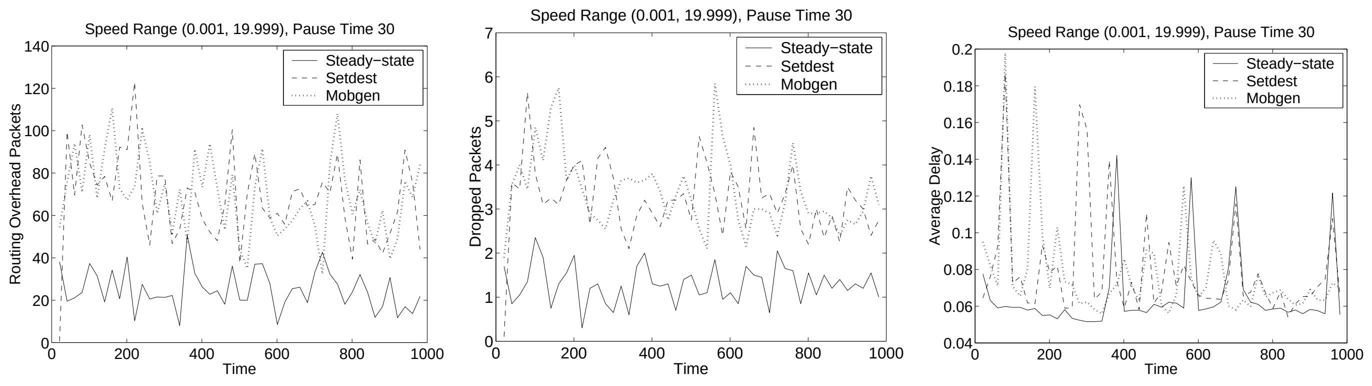

Fig. 7. Routing packets transmitted, dropped data packets, and average end-to-end delay for seconds 21 through 999 for three mobility models.

IV. Conclusions

Many published simulation results that compare mobile ad hoc network routing protocols use the random waypoint mobility model. To our knowledge, all of these published simulation results began the simulations with either all or (approximately) half of the nodes paused. In addition, only a few of the published simulation results discard an initial sequence of observations in the hope that the remaining values will closely represent the "steady-state" distribution of the RWM model. We show, however, that convergence of the average speed can take more than 1000 seconds of simulation time if the minimum speed is low. Furthermore, due to this slow convergence, the performance of a network protocol will vary systematically with time. In other words, the performance results are an artifact of how long a simulation executes.

If, however, the initial locations and speeds of the nodes are chosen from the stationary distribution, convergence is immediate, no data need be discarded, and performance results are reliable. We show how to implement a steady-state mobility model generator (*mobgen-ss*) for the RWM model. (Our *mobgen-ss* code is available at `http://toilers.mines.edu`.) Our simulation results stress the importance of using *mobgen-ss*, especially when the minimum speed is small or when a non-zero pause time is used. Since *mobgen-ss* is easy to use, we encourage the MANET community to begin using it for all their future simulations.

References

[1] C. Bettstetter. Smooth is better than sharp: A random mobility model for simulation of wireless networks. In *Proceedings of the Fourth ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'01)*, pages 19–27, 2001.

[2] C. Bettstetter, G. Resta, and P. Santi. The node distribution of the random waypoint mobility model for wireless ad hoc networks. Technical Report TUM-LKN 2002/01, Technische Universität München, Institute of Communication Network, 2002.

[3] C. Bettstetter and C. Wagner. The spatial node distribution of the random waypoint mobility model. In *Proceedings of the First German Workshop on Mobile Ad-Hoc Networks (WMAN), GI Lecture Notes in Informatics, P-11*, pages 41–58, 2002.

[4] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva. Multi-hop wireless ad hoc network routing protocols. In *Proceedings of the Fourth Annual ACM International Conference on Mobile Computing and Networking (MobiCom 1998)*, pages 85–97, 1998.

[5] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communications & Mobile Computing (WCMC)*, 2(5):483–502, 2002.

[6] T. Camp, J. Boleng, B. Williams, L. Wilcox, and W. Navidi. Performance comparison of two location based routing protocols for ad hoc networks. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom 2002)*, pages 1678–1687, 2002.

[7] C. Chiang and M. Gerla. On-demand multicast in mobile wireless networks. In *Proceedings of the IEEE International Conference on Network Protocols (ICNP)*, pages 262–270, 1998.

[8] W. Feller, editor. *An Introduction to Probability and its Applications, Vol II, 2nd ed.* John Wiley and Sons, 1970.

[9] J.J. Garcia-Luna-Aceves and M. Spohn. Source-tree routing in wireless networks. In *Proceedings of the 7th International Conference on Network Protocols (ICNP)*, pages 273–282, 1999.

[10] Marc Greis and The VINT Group. Tutorial for the network simulator - ns. http://www.isi.edu/nsnam/ns/tutorial/index.html. Page accessed June 27th, 2003.

[11] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark. Routing protocols for mobile ad-hoc networks - a comparative performance analysis. In *Proceedings of the Fifth Annual ACM International Conference on Mobile Computing and Networking (MobiCom 1999)*, pages 195–206, 1999.

[12] D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. In T. Imelinsky and H. Korth, editors, *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[13] D. Johnson, D. Maltz, and Y. Hu. The dynamic source routing protocol for mobile ad hoc networks. Internet Draft: draft-ietf-manet-dsr-09.txt, April 2003.

[14] Y. Ko and N.H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. In *Proceedings of the Fourth Annual ACM International Conference on Mobile Computing and Networking (MobiCom 1998)*, pages 66–75, 1998.

[15] W. Navidi and T. Camp. Stationary distributions for the random waypoint mobility model. Technical Report MCS-03-04, Colorado School of Mines, 2003.

[16] The Rice Monarch Project. The Rice monarch extensions to the ns simulator. http://www.monarch.cs.rice.edu/cmu-ns.html. Page accessed on August 14, 2002.

[17] E. Royer, P.M. Melliar-Smith, and L. Moser. An analysis of the optimum node density for ad hoc mobile networks. In *Proceedings of the IEEE International Conference on Communications*, pages 857–861, 2001.

[18] Scalable Network Technologies. Qualnet. http://www.scalable-networks.com. Page accessed on June 27th, 2003.

[19] J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful. In *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom '03)*, 2003.