Characterizing Network Dynamics in Mobile Wireless Networks

by

Jiaxuan Chen
Class of 2023

A thesis submitted to the
faculty of Wesleyan University
in partial fulfillment of the requirements for the
Degree of Bachelor of Arts
with Departmental Honors in Computer Science

Acknowledgements

First, I would like to thank my thesis advisor Professor Victoria Manfredi for advice and assistance in finishing the thesis. From the machine learning class to the networking tutorial, your guidance in classes equips me with proper knowledge and tools to complete this thesis. In our weekly meetings, we narrow down the research topics, set up procedural goals. I learn and grow through our discussion about network analysis, machine learning and most importantly, how to express the ideas and findings clearly, translating theories and results into readable paragraphs.

In addition, I would like to thank my academics advisors, Professor Dan Licata from Computer Science department, Professor Ruth Johnson from Biology department and Professor Gilbert Skillman from Economics department. Thank you for supporting me to explore all my academic interests boldly during my academic journey at Wesleyan. From course registration to research recommendations, your advice pushes me forward and encourages me to challenge myself with three distinct majors. I would also want to express my gratitude to my other research advisors, Professor Jorge Vasquez and Professor Richard Grossman in your guidance on economics researches in regards to misinformation spread and British option market. I also appreciate all other professors who provides the help and support during my years at Wesleyan that enables me to become who I am today.

Last but not least, I would like to thank my mom and dad to unconditionally believe in me and support my decision to study aboard in the US. Finishing my degree during the pandemic is not easy. It is uncertain and full of anxiety. During that special time, it was your video call every day that calmed me down and allowed me to focus on what I wanted to do. Now, I would like to present you with this thesis, a culmination of my studies at Wesleyan, a gift for you.

Abstract

Mobile ad hoc networks (MANETs) are a type of wireless network in which devices are mobile and are connected with each other via wireless links. To exchange traffic, these networks relying on devices forwarding traffic for each other, rather than using fixed network infrastructure like cellular towers or Internet routers. Applications of MANETs range from disaster rescue, to national defense systems, to home automation, to robotics. Routing and forwarding of traffic in a MANET is difficult, however, because the network topology is changing over time. Metrics that are able quantify how the network topology is changing can be used to make better routing decisions in these networks. In this thesis, we investigate four metrics to quantify how the connectivity and predictability of the network topology changes over time in a MANET. The specific metrics that we consider are the average node degree, the probability that a path exists, the average link-up entropy, and the average link-down entropy. In simulation, we evaluate how these metrics vary as a function of different mobility models and different network settings. Our results show that these metrics, when computed for different mobility models, show common trends as a function of node transmission range and node speed.

Contents

1 Introduction

1.1 Mobile Ad hoc Network

Mobile ad hoc networks (MANETs) [4] are a type of wireless networks in which devices are mobile and are connected with each other via wireless links. To exchange traffic, these networks relying on devices forwarding the traffic for each other, rather than using infrastructure like cellular towers or Internet routers. Such networks can be configured as a temporary network. Applications of MANETs can range from disaster rescue [3], to national defense systems [12, 7], to home automation [8], to robotics [6]. Examples of MANETs include vehicular ad hoc networks which could involve autonomous vehicles moving in different streets communicating with each other to avoid obstacles and traffic congestion, and smartphone ad hoc networks that use WIFI and bluetooth to create a wireless network without relying on cellular communication.

Abstractly a MANET can be represented as a graph that is changing over time. The nodes in the graph are moving according to some mobility, causing the temporal changes in the graph topology. Such mobility can be abstractly represented using what is called a mobility model. The goal of a mobility model is to mimic a real-life scenario: people with mobile devices move around to various locations in an environment with different speeds and take different routes over time.

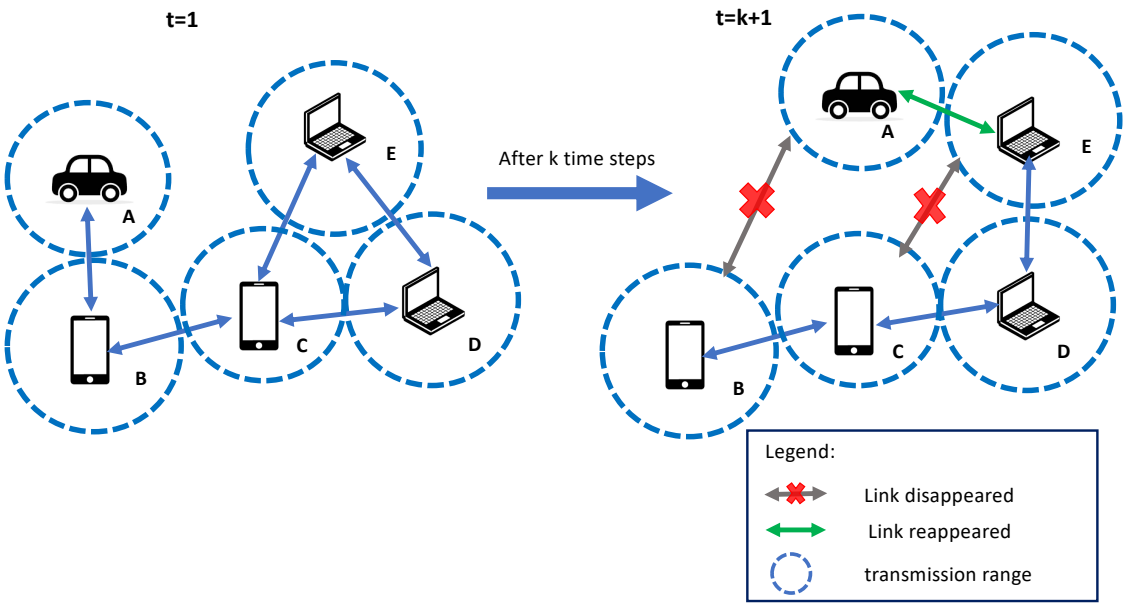


Figure 1: Mobile Ad Hoc Network

1.2 Routing and Forwarding in MANETs

Routing and forwarding of traffic in a MANET [1] is difficult because the network topology is changing over time. Since MANETs are built over mobile devices whose locations are changing over time since the devices are moving , it is difficult for the packet to be sent from one device to another because finding a stable or predictable path is difficult. For example, consider a car trying to send the message through a neighboring device such as a mobile phone or a laptop. The packet may not be able to reach its destination because the car or the mobile phone may move to a different location and be unable to connect properly to transfer the message from one device to the other (see Figure 1). In others words, we cannot know for sure who the neighbours of the mobile devices will be over time because of the device's mobility. This uncertainty causes the routing and forwarding of packets to be unpredictable.

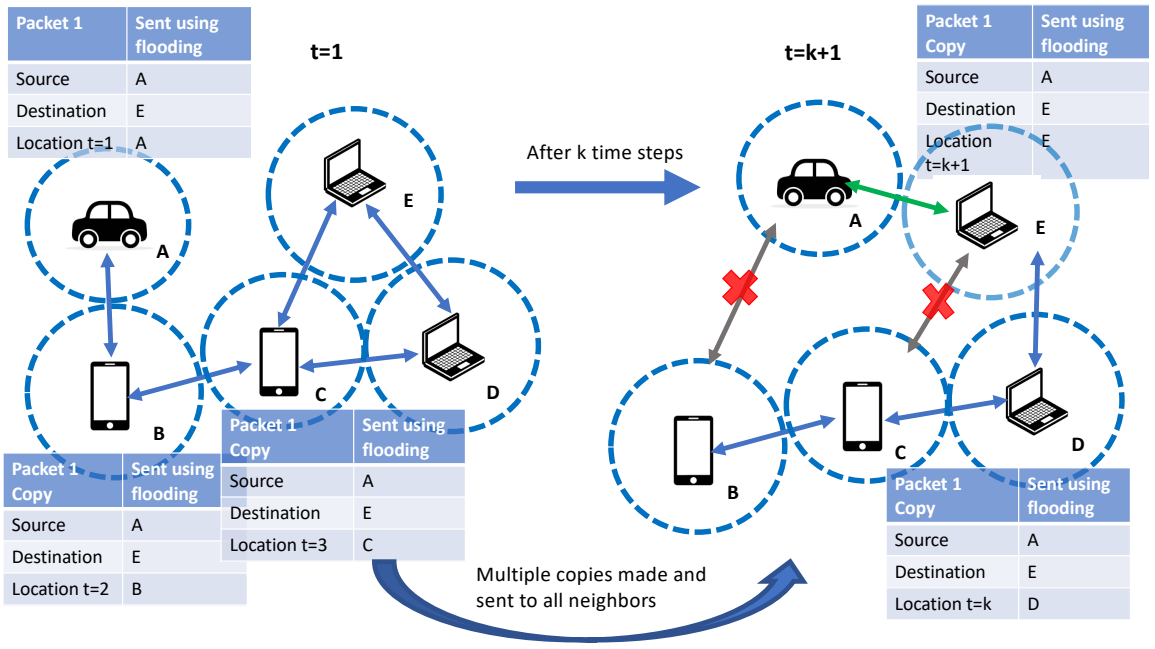


Figure 2: Flooding Transmission

Additionally, a network may also face an efficiency problem when devices try to send a packet. In particular, there are two “brute-force" approaches that devices may use to deliver a packet to its destination in a MANET. Devices can send packets to all current neighbors of a device to ensure that the packet is delivered quickly by making multiple copies of the message (flooding). However, imagine that there are thousands of devices in the network. In this scenario, the routing approach of flooding will be extremely costly and waste a lot of bandwidth and may even cause congestion due to the large amount of traffic sent over the network (as illustrated in Figure 2).

An alternative approach to getting packets to their destinations is for the source devices of the packets to hold on to their packets and hope that destination devices will move into the devices transmission range so the a source can send the packet directly to the destination (direct transmission). This can also be inefficient, taking awhile for the packet to be delivered, given the uncer-

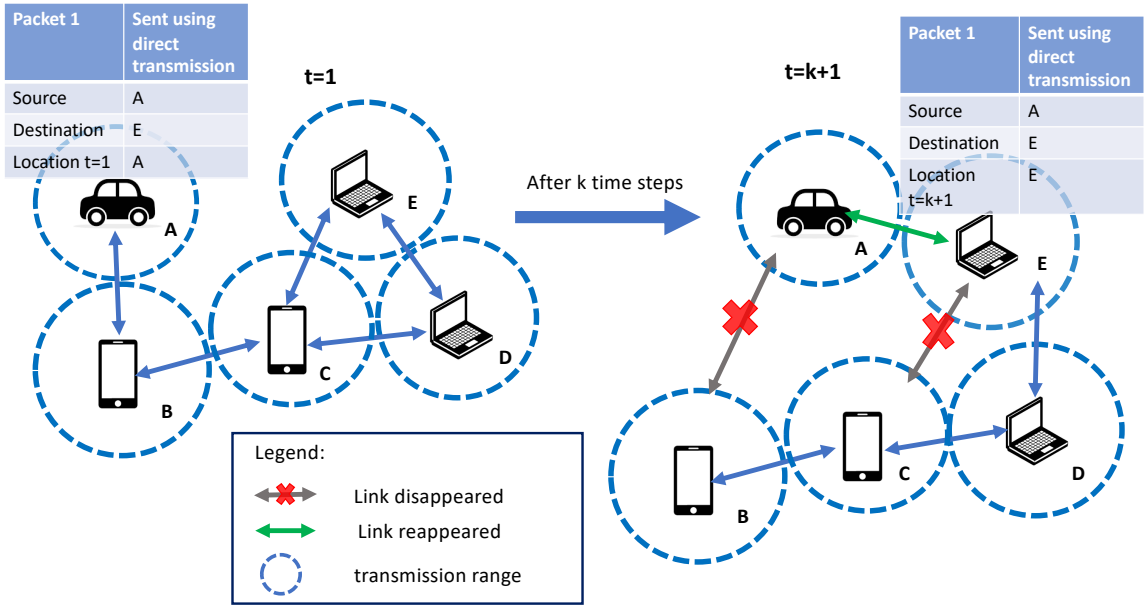


Figure 3: Direct Transmission

tainty of device movement in the MANET. If the destination device happens to move in an unexpected direction and take a long time before being with transmission range of the source devices, the packet delivery time will also be long. Thus, even though direct transmission incurs less network traffic, it is very inefficient in terms of delay (as shown in Figure 3).

Since there are many different ways of routing depending on how the network topology changing (flooding or sending multiple copies of a packet, waiting for the packet's destination, or collecting routing state and routing over a path), it is helpful to be able to detect things about the network topology that tell us what strategy is best in a given network scenario.

1.3 Project Purpose

This thesis aims to explore approaches to characterize device mobility in a MANET with the eventual goal of using these characterizations to improve routing. We explore different mobility models and propose metrics to characterize how mobility varies in these models. By identifying key metrics that characterize mobility, we get closer to being able to map any arbitrary device movement to a common feature space, allowing routing decisions to be made on that feature space rather than requiring individualized approaches to routing for different kinds of mobility.

The thesis goals are as follows.

1. Design metrics that could be useful for distinguishing when different routing paradigms should be used.

2. Design metrics that could be useful for making routing decisions within a routing paradigm, such as deciding the number of packet copies to send or deciding which node is the best next hop for packet.

In this thesis, we evaluate the utility of four network metrics computed over space and time in different MANETs with different mobility characteristics. The metrics we explore are average node degree, probability path exists, average link-up entropy, and average link-down entropy. The code repository for this project is here.

2 Temporal Graph Metrics

In this section, we identify and describe several temporal graph metrics to model a network graph topology that is changing over time.

2.1 Network Representation as a Graph

We focus here in this work on using undirected graphs to abstract a mobile network topology. We abstract a network as a sequence of graphs over time where $G_t = (V_t, E_t)$ is the state of the network graph at time $0 \leq t \leq T$. We define $N_t = |V_t|$ as the set of nodes in the graph at time t and E_t as the set of undirected edges present in the network at time t. In our simulations in the next section, we set $T = 900$ seconds. Since the set of nodes does not change over time, we use $V = V_t$ and then correspondingly the number of nodes is $N = |V|$.

2.2 Network Connectivity

In this section, we describe two metrics to quantify how well-connected the topology of a network is. Having estimates of the network connectivity is important for understanding how best to route and forward traffic because if the network is very poorly connected, then it may make sense to not try to route, but to instead just make copies of packets whenever two nodes meet. If the network is well-connected, then routing makes more sense to do.

2.2.1 Average Node Degree

Average node degree is an useful metric to measure the overall connectivity between nodes in a network over time. Since we are measuring the overall connectivity of the network, we take the average of node degrees for each node in the network over time. Given the mobility of nodes in the network, we take snapshots of network over time and make an average over averaged node degrees of the network in each snapshot. To calculate the average node degree, we first need to compute the degree (D_i) for each node, defined as follows.

$$D_i = \text{total edges incident to node } i$$

Then we sum up node degrees D_i for each node i in the network and divide it by total number of nodes N in the network, to get the average node degree in the network ($\bar{D}$).

$$\bar{D} = \sum_{i=1}^{N} \frac{D_i}{N} \tag{1}$$

While here we use just a simple average, a weighted average of some kind could also be used.

2.2.2 Probability that a Path Exists

The *probability that a path exists* measures how likely it is that a contemporaneous end-end path exists between any two nodes in the network. To calculate the probability that a path exists, we first need to know the total number of possible (shortest) paths can exist between nodes for the graphs at each time step t. If there are have N nodes, then there are at most $\frac{1}{2}N \times (N-1)$ possible source and destination pairs, and hence paths. After that, we compute the number of (shortest) paths ($N_{shortest}$) that exist in the network G_t using an all-pair shortest paths algorithm. Then we compute the probability that a path exists at time t (P_t), in the graph G_t, as follows.

$$P_t = \frac{N_{shortest}}{2N \times (N-1)}$$

To get the average of the probability that a path exists ($\overline{P}_{path}$) from time t to T, we have:

$$\overline{P} = \sum_{t=1}^{T} \frac{P_t}{T} \tag{2}$$

2.3 Network Predictability

In this section, we describe two entropy-based metrics to quantify how predictable the changes in a network's topology are. Being able to measure predictability is important for routing because if the network topology is very

unpredictable, actually computing and maintaining routing state may not be useful: by the time state is computed it will be out-of-date. Instead flooding or multi-copy routing strategies may be more useful to use. Both of the entropy-based metrics that we look at rely on comparing how different the state of network graph is at different points of time: one metric looks at links that have disappeared over time and the other looks at links that have appeared over time.

2.3.1 Average Link-up Entropy

We evaluate one metric, termed *average link-up entropy* previously proposed in [10], based on computing the conditional entropy and described again here for clarity. Average link-up entropy is computed by comparing the state of the network graph at one point in time with future points in time, focusing on the links that were originally up and are now transitioning down in those future points in time.

Let time t be when nodes last collected information about the state of the network and let G_t be the state of the network graph at time t. This state is the link-up state as it indicates which links are actually up in the network time t. Let time $t+k$ be some time in the future and let G_{t+k} be the state of the network graph at time $t+k$. The key point is that the network state at time $t+k$ may or may not be similar to the network state at time t depending on how the graph topology has changed in the k timesteps since t. Our goal is to quantify how this state has changed.

To compute average link-up entropy, we focus on the subset of edges in G_{t+k} that were up in G_t, i.e., $E^{up}_{t+k|t} \subseteq E_t$. We define $g_{t+k|t}$ to be a binary random

variable indicating whether an arbitrary link is up or down in the graph G_{t+k} relative to the links that were present (i.e., up) in the graph G_t. The probability that an arbitrary link that was up at time t is still up at time $t+k$ is given by the following.

$$P(g_{t+k} = up | g_t = up) = \frac{|E^{up}_{t+k|t}|}{|E_t|} \quad (3)$$

The probability that an arbitrary link that was up at time t is down at time $t+k$ is given by the following.

$$P(g_{t+k} = down | g_t = up) = \frac{|E_t| - |E^{up}_{t+k|t}|}{|E_t|} \quad (4)$$

We can then define *Link-up entropy*, $H_{up}(g_{t+k}|g_t)$ as follows.

$$H_{up}(g_{t+k}|g_t) = -\sum_{y=up,down} P(g_{t+k} = y|g_t = up) \log P(g_{t+k} = y|g_t = up) \quad (5)$$

We then average the link-up entropy over all timesteps in an interval Δ as follows.

$$h_{up,\Delta} = \frac{1}{\Delta} \sum_{k=0}^{\Delta-1} \mathcal{H}_{up,t+k|t} \quad (6)$$

where

$$\mathcal{H}_{up,t+k|t} = \begin{cases} H_{up}(g_{t+k}|g_t) & \text{if } t \neq 1 \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

Note we use 0 in the above equation, rather than 1 like in [10] to convey total knowledge about the state of the network at the first time step in each interval Δ, specifically which links are up or down in the network. Average link-up entropy thus takes the average over all of the link-up entropies computed in the Δ intervals in a simulation trace with T timesteps.

2.3.2 Average Link-down Entropy

We introduce a complement to average link-up entropy, which we term *average link-down entropy* to consider those links that are transitioning from down to up in the network over time.

To compute average link-down entropy, we focus on the subset of edges in G_{t+k} that were down in G_t, i.e., $E^{down}_{t+k|t} \subseteq E^C_t$. We define q_{t+k} to be a binary random variable indicating whether an arbitrary link is up or down in the graph G_{t+k} relative to the links that were *not* present (i.e., down) in the graph G_t.

To do this, we first compute the link-down entropy, as follows.

$$H_{down}(q_{t+k}|q_t) = -\sum_{y=up,down} P(q_{t+k} = y|q_t = down) \log P(q_{t+k} = y|q_t = down) \quad (8)$$

Then we compute the average link-down entropy as follows by taking the average of

the link-down entropies over all timesteps in an interval Δ as follows.

$$h_{down,\Delta} = \frac{1}{\Delta} \sum_{k=0}^{\Delta-1} \mathcal{H}_{down,t+k|t} \tag{9}$$

where

$$\mathcal{H}_{down,t+k|t} = \begin{cases} H_{down}(q_{t+k}|q_t) & \text{if } t \neq 1 \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

Note we use 0 in the above equation, rather than 1 like in [10] to convey total knowledge about the state of the network.

2.4 Example Average Link-Up Entropy Calculation

To demystify the calculation of average link-up entropy (and average link-down entropy), we show in Figures 4 (and 5) a simplified graph representation of a network with $N = 5$ nodes and compute the average link-up (and link-down entropies) using a small update interval ($\Delta = 3$) to give more insight.

First, we need to calculate the number of edges that are present in the network (i.e., up edges) and the number of edges that could be present but are not currently present in the network (i.e., down edges) at the initial timestep, at time $t = 1$. The total possible number of edges in the network in Figures 4 and 5 is $\frac{N \times (N-1)}{2} = \frac{5 \times 4}{2} = 10$. The number of up edges is 5 when $t = 1$. Then the number of down edges at $t = 1$ is $10 - 5 = 5$. This is shown in the tables underneath the $t = 1$ network graph representations in Figures 4 and 5.

Figure 4 illustrates how we compute average link-up entropy. In Figure 4, there are two edges present at time $t = 1$ that have disappeared at $t = 2$ (illustrated as red dotted lines). There are no new edges created from $t = 1$ to $t = 2$. To compute the link-up entropy at each time step, we first need to get the Probability that edges that

are up given they are up in the first time step ($t = 1$) of the Δ interval ($P(up \mid up)$) and Probability that edges that are down given they are up in the first time step ($t = 1$) of the Δ interval ($P(down \mid up)$). In this case, we have the following for $t = 2$.

$$\begin{aligned} P(up \mid up) &= \frac{\text{Up edges at } t = 2 \text{ given they are up at } t = 1}{\text{Up edges at t=1}} \\ &= \frac{5 - 2}{5} \\ &= \frac{3}{5} \end{aligned}$$

$$\begin{aligned} P(down \mid up) &= \frac{\text{Down edges at } t = 2 \text{ given they are up at } t = 1}{\text{Up edges at } t = 1} \\ &= \frac{2}{5} \end{aligned}$$

Then we can calculate the link-up entropy for $t = 2$ and $t = 3$ by comparing with the $t = 1$ graph. Since the graph at time $t = 1$ is comparing we itself, we set its up entropy to 0 (no change in the network at all). In the bottom part of Figure 4, there is a summarized calculation for computing link-up entropy for $t = 2$ ($H_{up}(t = 2 \mid t = 1)$). The next equation writes out the full calculation for computing link-up entropy as given in Figure 4.

$$\begin{aligned} H_{up}(t = 2 \mid t = 1) &= -P(up \mid up) \cdot \log(P(up \mid up)) - P(down \mid up) \cdot \log(P(down \mid up)) \\ &= -\frac{3}{5} \cdot \log(\frac{3}{5}) - \frac{2}{5} \cdot \log(\frac{2}{5}) \\ &\approx 0.292 \end{aligned}$$

Similarly, since there are no other edges disappearing relative to the original graph at time $t = 1$ then the graph at time $t = 3$ should have the same link-up entropy as

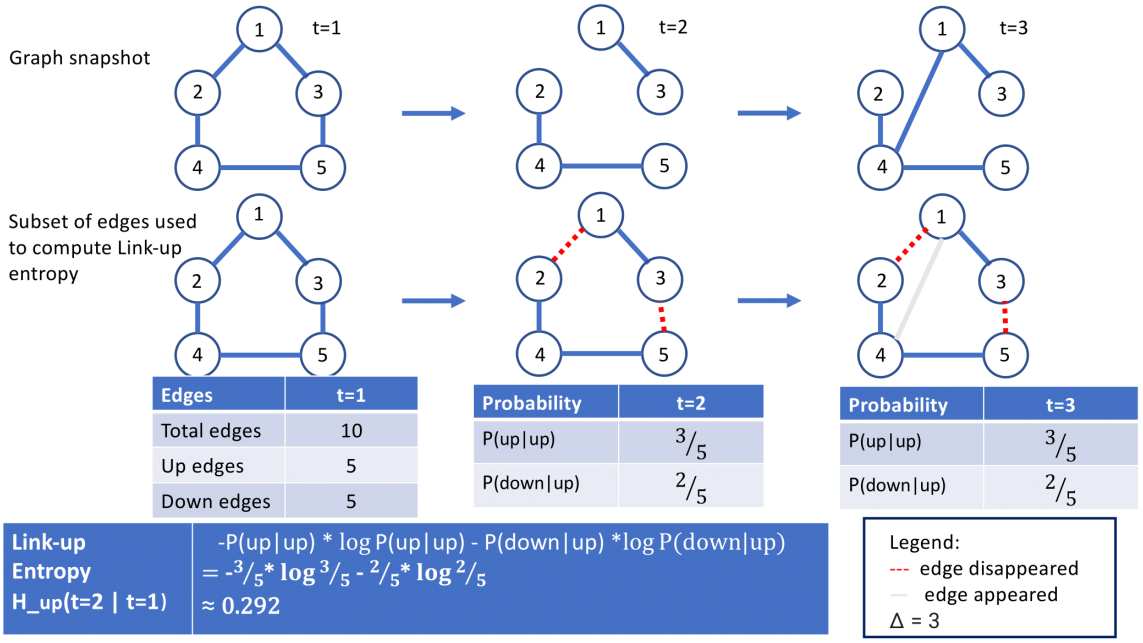


Figure 4: Average link-up entropy: example computation over time

the graph at time $t = 2$. That is, $H_{up}(t = 3 \mid t = 1) \approx 0.292$. Finally, we average the link-up entropy of the three timesteps to compute the average link-up entropy of this Δ interval ($h_{up,\Delta=3}$). In our simulations in Section 4, we generated 900 time steps (where one timestep corresponds to one second) and use 3 different Δ intervals ranges (of 20,50, and 200 seconds) for averaging the link-up entropies.

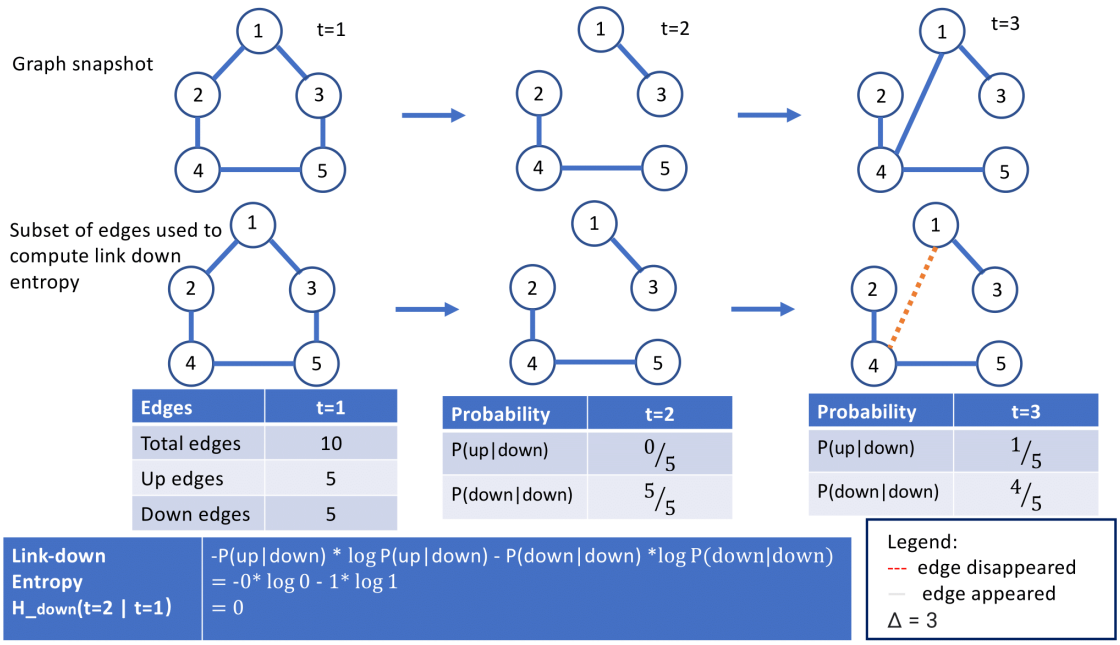


Figure 5: Average link-down entropy: example computation over time

$$
\begin{aligned}
h_{up,\Delta=3} &= \frac{1}{\Delta}\sum_{k=0}^{\Delta-1}\mathcal{H}_{t+k|t} \\
&= \frac{1}{\Delta}\sum_{k=0}^{2}\mathcal{H}_{t+k|t} \\
&= \frac{1}{3}\cdot\left(H_{up}(t=1\mid t=1)+H_{up}(t=2\mid t=1)+H_{up}(t=3\mid t=1)\right) \\
&= \frac{0+0.292+0.292}{3} \\
&\approx 0.195
\end{aligned}
$$

To compute the average link-up entropy over the Δ intervals, we first averaged within the each Δ interval and then we averaged across the averages.

2.5 Example Average Link-down Entropy Calculation

Figure 5 illustrates how we compute the average link-down entropy. To calculate the average link-down entropy, now our concern is how many edges are added to the later network graphs at later timesteps given the graph at the first time-step. At time $t = 2$, there are no edges added. Similarly, there are no edges added at time $t = 3$. To compute the link-down entropy for each time step, we first need to get the probability that edges that are up given they are down at the first time step ($t = 1$) of the Δ interval ($P(up \mid down)$) and the probability that edges that are down given they are down at the first time step ($t = 1$) of the interval ($P(down \mid down)$). In this case, we have for $t = 2$, the following.

$$\begin{aligned} P(up \mid down) &= \frac{\text{Up edges at t=2 given they are down at t=1}}{\text{Down edges at t=1}} \\ &= \frac{0}{5} \\ &= 0 \end{aligned}$$

$$\begin{aligned} P(down \mid down) &= \frac{\text{Down edges at t=2 given they are down at t=1}}{\text{Down edges at t=1}} \\ &= \frac{5}{5} \\ &= 1 \end{aligned}$$

Then for $t = 3$ we have the following.

$$\begin{aligned} P(up \mid down) &= \frac{\text{Up edges at t=3 given they are down at t=1}}{\text{Down edges at t=1}} \\ &= \frac{1}{5} \end{aligned}$$

$$
\begin{aligned}
P(down \mid down) &= \frac{\text{Down edges at t=3 given they are down at t=1}}{\text{Down edges at t=1}} \\
&= \frac{5-1}{5} \\
&= \frac{4}{5}
\end{aligned}
$$

We then use these probabilities to calculate the link=down entropy for each time step and average them to get the average link-down entropy for this interval (see also the $t = 2$ link-down entropy calculation shown in Figure 5).

$$
\begin{aligned}
H_{down}(t = 2 \mid t = 1) &= -P(up \mid down) \cdot \log(P(up \mid down)) - P(down \mid down) \cdot \log(P(down \mid down)) \\
&= -0 \cdot \log 0 - 1 \cdot \log 1 \\
&= 0
\end{aligned}
$$

$$
\begin{aligned}
H_{down}(t = 3 \mid t = 1) &= -P(up \mid down) \cdot \log(P(up \mid down)) - P(down \mid down) \cdot \log(P(down \mid down)) \\
&= -\frac{1}{5} \cdot \log(\frac{1}{5}) - \frac{4}{5} \cdot \log(\frac{4}{5}) \\
&\approx 0.217
\end{aligned}
$$

$$
\begin{aligned}
h_{down,\Delta=3} &= \frac{1}{\Delta}\sum_{k=0}^{\Delta-1}\mathcal{H}_{t+k|t} \\
&= \frac{1}{\Delta}\sum_{k=0}^{2}\mathcal{H}_{t+k|t} \\
&= \frac{1}{3}\cdot\left(H_{down}(t=1 \mid t=1) + H_{down}(t=2 \mid t=1) + H_{down}(t=3 \mid t=1)\right) \\
&= \frac{0+0+0.217}{3} \\
&\approx 0.097
\end{aligned}
$$

Similarly, we average over the down entropy in each Δ interval in the simulation to get the averaged down entropy across Δ interval ranges of 20, 50, and 200 seconds.

3 Mobility Models

In this section, we overview the different mobility models we use in our simulations to evaluate how well our different network metrics are able to characterize device mobility.

3.1 Modeling Device Mobility

To simulate the users in a Mobile Ad Hoc Network (MANET), different mobility models can be used to predict the movement of users' behaviors within the network. Since a mobility model controls the moving speed and direction of the movements of nodes over time, the performance of routing and the networks' connectivity will vary as a function of the mobility dictated by the models. The mean speed of nodes and transmission range parameters are altered in the simulations in order to compare the connectivity and predictability of movement across three models:

Steady State Random Waypoint, Original Gauss-Markov, and Manhattan Grid. The implementation of the simulations are achieved using BonnMotion [2], a Java-based mobility scenario generation and analysis tool to provide the proper node files which will later be used in the calculation of the evaluation metrics.

In the next sections, we will first go through the mobility models used in the simulations and then describe the implementation and parameter settings within each of the models in the BonnMotion implementation.

3.2 Steady-State Random Waypoint Mobility Model

For the Steady State Random Waypoint model, initially each node will be given a starting point in the probability distribution of locations as well as a destination. The speed is chosen from an interval uniformly and gets updated after the node reaches its destination and heads to a new destination (i.e., new waypoint). In a real life scenario, you can imagine that a person is walking from their home to multiple stores at different locations with their mobile phone and traveling via different kinds of transportation. The phone itself will be considered as a node in the Mobile Ad Hoc Network(MANET) travelling at different speed to different destinations over time. Because of the variation of speed and location throughout the simulation and particularly during the initial part of the simulation, the network performance can be inconsistent overtime. That is why we use introduce the steady state version of the Random Waypoint model in the simulation so that it will be yield sound results for the evaluation metrics we investigate. To additionally ensure statistical soundness, device pausing is not permitted in our simulations and the initial transient period is cutoff to avoid including result from the initial high variability phase of the simulation. [11]

3.3 Original Gauss Markov Mobility Model

For the original Gauss-Markov Model, the movement of the mobile devices is updated based on the past and current locations and speeds of the devices. Unlike the Random Waypoint model where everything is random and ""memory-less"", the Gauss-Markov model introduces the weighting of the past location and noise like the Gauss-Markov process to update the next location and speed of each nodes. In the 2D scenario, the location and speed of a mobile device is represented by random vectors $[s_n^x, s_n^y]$ and $[V_n^x, V_n^y]$. The weighting, standard deviation and mean of the distribution can be represented as $\overline{\alpha} = [\alpha_n^x, \alpha_n^y]$, $\overline{\sigma} = [\sigma_n^x, \sigma_n^y]$ and $\overline{\mu} = [\mu_n^x, \mu_n^y]$. Then we have the expression for the Original Guass Markov update as follows. [5]

$$V_n = \overline{\alpha} \odot V_{n-1} + (1 - \overline{\alpha}) \odot \overline{\mu} + \overline{\sigma} \odot \sqrt{1 - \overline{\alpha}^2} \odot W_{n-1}$$

Note: $\odot$ is element-wise multiplication of the vector

In this update, W_{n-1} is the noise of the processes uncorrelated with V_n. We can simplify the expression by assuming we have same level of weighing (α) over time:

$$V_n = \overline{\alpha} V_{n-1} + (1 - \overline{\alpha}) \odot \overline{\mu} + \overline{\sigma} \odot \sqrt{1 - \overline{\alpha}^2} \odot W_{n-1}$$

The transmission range and mean speed of nodes are varied in our simulations to vary the probability of a path existing, node degree, and entropy metrics. For the sake of the simplicity, pausing of devices is again turned off when generating the mobility traces for simulation and the initial transient period is cutoff is set to avoid including result from the initial high variability phase of the simulation. [9]

3.4 Manhattan Grid Mobility Model

For the Manhattan Grid model, it is often used to model vehicle-based mobility where devices can only move along the grid-like predefined paths. The topology of the space is divided by the numbers of blocks (like the city of Manhattan). The mobility of nodes can be affected by setting their mean speed, the probability to change speed at the current position, and the probability to turn at an intersection of the grid.

To additionally ensure statistical soundness, device pausing is not permitted in our simulations and the initial transient period is cutoff to avoid including result from the initial high variability phase of the simulation.

For the sake of the simplicity, pausing is ignored in our simulations and the initial transient period cutoff is set to avoid including result from the initial high variability phase of the simulation. In addition to transmission range and mean speed, the turn probability is also varied in our network simulations to vary the different values of the metrics we propose to use to characterize network dynamics.

4 Simulations

In this section, we describe our simulation results. Our goal in these simulations is to evaluate how the values of the different metrics we propose (probability that a path exists, node degree, average link-up entropy, and average link-down entropy) vary as a function of different mobility models. We first describe how we perform our simulations then overview our results in detail.

4.1 Methods

4.1.1 Generating Traces for Mobility Models

To generate mobility traces to analyze in our simulations, we used BonnMotion [2], a mobility scenario generation and analysis tool. For each of the mobility models we consider, we generate traces of 900 seconds, after having cut off the initial 3600 seconds. Each trace contains 100 nodes moving around a 500m $\times$ 500m area.

We next describe the BonnMotion commands that we use to generate the mobility traces of the Steady-State Random Waypoint, Original Gauss-Markov, and Manhattan Grid mobility models.

All scenarios use the following flags.

- n number of nodes, set to 100
- x width of the simulation area, set to 500
- y height of the simulation area, set to 500
- i number of seconds to skip, set to 3600
- d scenario duration, set to 900

In our simulations, we set the number of nodes we use to 100 (-n 100), the simulation area to 500m $\times$ 500m (-x 500 and -y 500), the number of seconds to skip at the beginning of a simulation to 3600 seconds and the scenario duration after the skipped part to be 900 seconds.

After we set the parameters for the scenario that we use to run simulation, we then choose the mobility model that we plan to generate traces. For each of the mobility models that we use, there are model-specific parameters that we need to additionally

set to generate the traces.

For the Steady State Random Waypoint Mobility Model we use the following parameters.

– o speed mean, altered from 1 to 3

The commands that we use to generate mobility traces for the Steady State Random Waypoint Mobility Model for our simulations are as follows.

```
./bm -f filename SteadyStateRandomWaypoint -n 100 -d 900 -i 3600 \
        -x 500 -y 500 -o 1
./bm -f filename SteadyStateRandomWaypoint -n 100 -d 900 -i 3600 \
        -x 500 -y 500 -o 2
./bm -f filename SteadyStateRandomWaypoint -n 100 -d 900 -i 3600 \
        -x 500 -y 500 -o 3
```

For the Original Gauss Markov Mobility Model we use the following parameters.

– a average mean speed, altered from 1 to 3

– w weight, set to 0.3

– q update frequency, set to 30

– s velocity standard deviation, set to 0.5

The commands that we use to generate mobility traces for the Original Gauss Markov Mobility Model for our simulations are as follows.

```
./bm -f filename OriginalGaussMarkov -n 100 -d 900 -i 3600 \
```

```
          -x 500 -y 500 -a 1 -w 0.3 -s 0.5 -q 30
./bm -f filename OriginalGaussMarkov -n 100 -d 900 -i 3600 \
          -x 500 -y 500 -a 2 -w 0.3 -s 0.5 -q 30
./bm -f filename OriginalGaussMarkov -n 100 -d 900 -i 3600 \
          -x 500 -y 500 -a 3 -w 0.3 -s 0.5 -q 30
```

For the Manhattan Grid Mobility Model we use the following parameters.

- t turn probability, altered from 0.1 to 0.9
- c speed change probability, set to 0.3
- m mean speed, altered from 1 to 3
- s speed standard deviation, set to 0.5
- o maxpause, set to 0

The commands that we use to generate mobility traces for the Manhattan Grid Mobility Model for our simulations are as follows.

```
for mean speed m=1
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 1 -s 0.5 -t 0.1
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 1 -s 0.5 -t 0.3
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 1 -s 0.5 -t 0.5
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 1 -s 0.5 -t 0.9

for mean speed m=2
```

```
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 2 -s 0.5 -t 0.1
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 2 -s 0.5 -t 0.3
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 2 -s 0.5 -t 0.5
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 2 -s 0.5 -t 0.9

for mean speed m=3
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 3 -s 0.5 -t 0.1
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 3 -s 0.5 -t 0.3
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 3 -s 0.5 -t 0.5
./bm -f filename ManhattanGrid -n 100 -d 900 -i 3600 \
     -x 500 -y 500 -o 0 -c 0.3 -m 3 -s 0.5 -t 0.9
```

4.1.2 Setting Device Transmission Ranges

After we generate the mobility traces for each model, we need to use an another setting in the BonnMotion code, *InRangePrinter*, to tell us whether two nodes are connected in the network. Whether two nodes are connected depends on the transmission range we decide to use in a given trace. This is because we need information about the edges and graph connectivity in order to calculate the metrics mentioned in Section 2. The *InRangePrinter* has two modes: interval mode, which prints

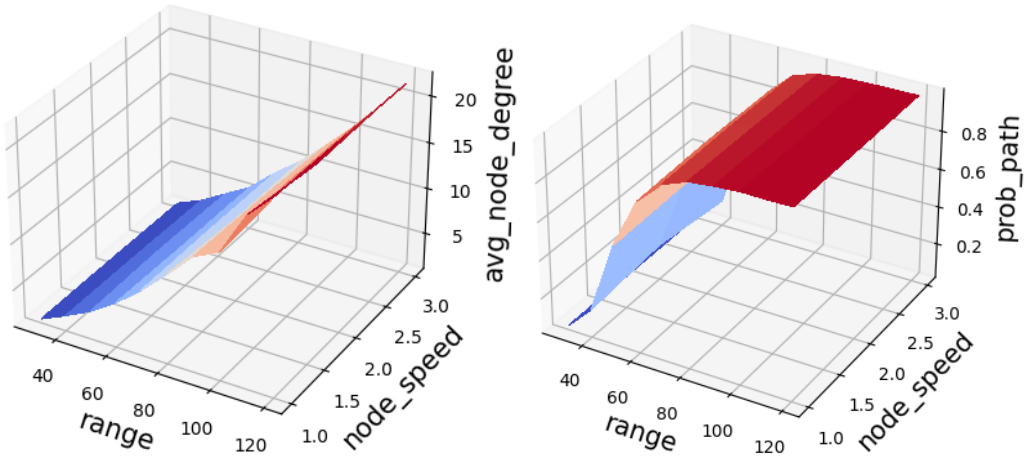
out information about graph connectivity given the indicated interval periodically, and range mode, which prints out information only when a node leaves or enter the transmission range of another node. In this work, we use the interval mode and set the interval flag equal to 1.0 second in order to keep track of network connectivity over time. In other words, we keep track of the edges in the network for every time step. For the $InRangePrinter$ we use the following parameters:

- f filename of the files that contains generated traces in the scenarios
- n number of nodes, set to 100
- l interval, set to 1.0
- r transmission range, altered from 30 to 120

The $InRangePrinter$ takes in the trace files generated by the mobility model simulations and then converts them into text files which contain 0s and 1s where connected corresponds to 1 and not connected corresponds to 0. From there, we parse the text file to organize the data into dataframes of 0s and 1s that have two-pair nodes tuple as headers ($[(0,1),(1,2),\ldots,(99,100)]$) via Python Pandas and Numpy modules.

The commands that we use to generate the text files that contain in-range information for our simulations are as follows.

```
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 30
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 40
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 50
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 60
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 70
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 80
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 90
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 100
```



(a) Average node degree, $\bar{D}$ (b) Probability path exists, P_{path}

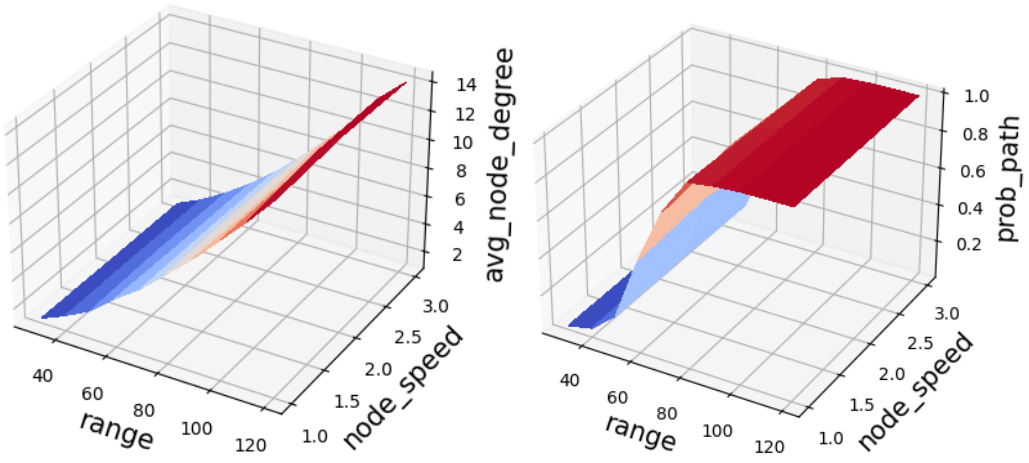
Figure 6: For the steady-state random waypoint mobility model, we show how average node degree and the probability that a path exists varies as a function of transmission range and node speed.

```
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 110
InRangePrinter -f traces\_filename -n 100 -l 1.0 -r 120
```

The last parameter refers to the transmission range and we set this to be 30m, 40m, 50m, 60m, 70m, 80m, 90m, and 100m.

4.2 Simulation Results

For each of the mobility models considered, we show the how each metric changes over time and varies according to different parameters of the mobility models. In the following sections, we discuss each of the metrics in Section 2 in turn.



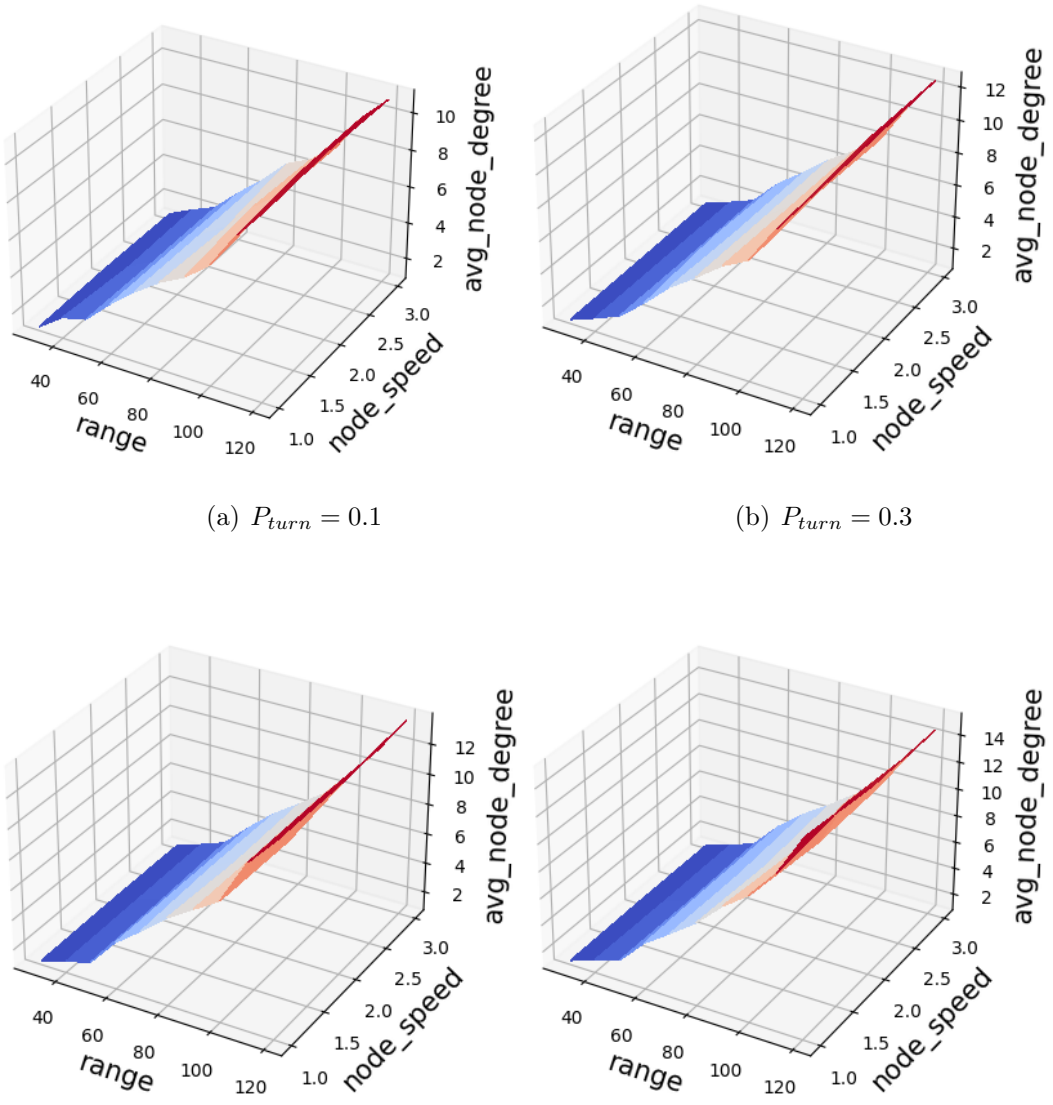
(a) *avg_node_degree* (b) *prob_path*

Figure 7: For the original Gauss-Markov mobility model, we show how average node degree and the probability that a path exists varies as a function of transmission range and node speed.

4.2.1 Average Node Degree

In this section, we examine how the average node degree, $\overline{D}$, changes as a function of the node transmission range and node speed. Figure 6(a) shows how $\overline{D}$ changes for the Steady-State Random Waypoint Mobility Model, while Figure 7(a) shows how $\overline{D}$ changes for the Original Gauss-Markov Mobility Model, and Figure 8 changes for the Manhattan Grid Mobility Model.

For the different mobility models, we see that despite the variation in mobility, that for all mobility models that as the transmission range increases, the average node degree also increases in the network. We observe that node speed does not have a significant impact on $\overline{D}$. In Figure 8 for the Manhattan Grid Mobility Model, we additionally vary the turn probability at intersections, P_{turn}, from a low probability of turning of $P_{turn} = 0.1$ to a high probability of turning of $P_{turn} = 0.9$. When we increase this turn probability we observe that the average node degree gets higher in the end of simulation.



(a) $P_{turn} = 0.1$

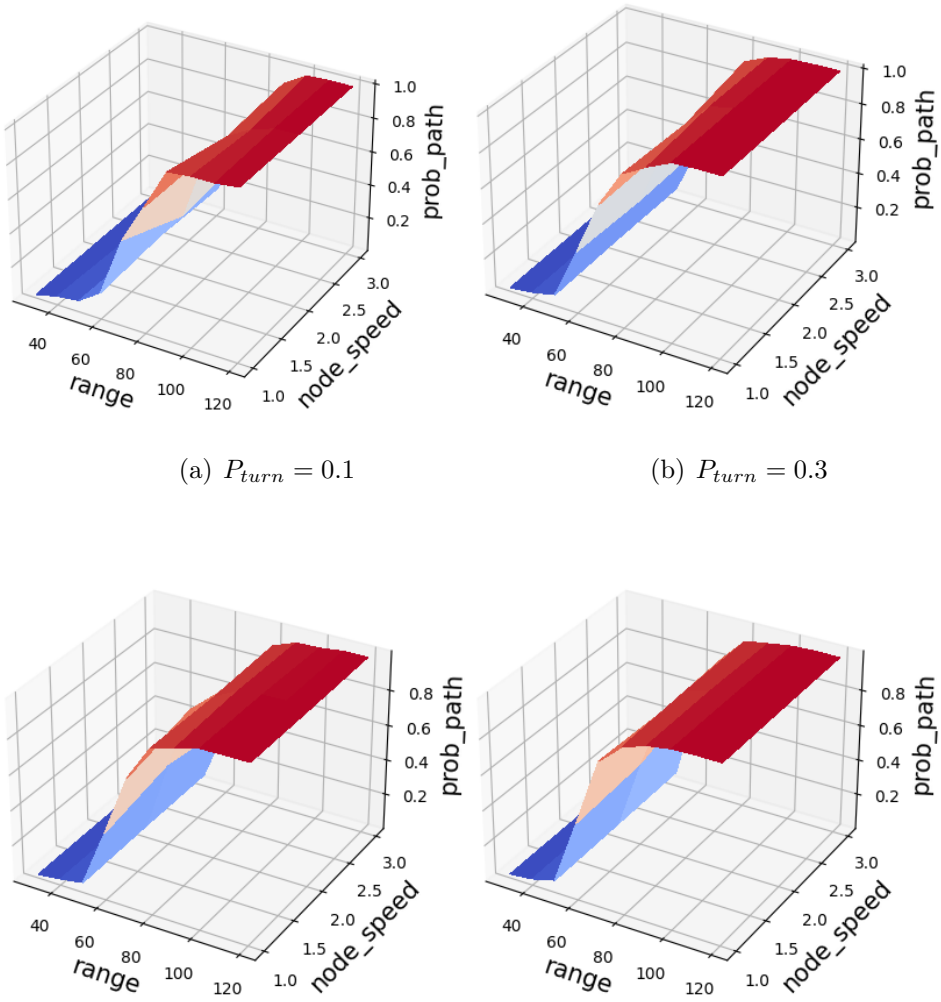
(b) $P_{turn} = 0.3$

(c) $P_{turn} = 0.5$

(d) $P_{turn} = 0.9$

Figure 8: For the Manhattan grid mobility model, we show how average node degree varies as a function of transmission range and node speed. We also vary the turn probability from 0.1 to 0.9 for the Manhattan grid mobility model only.

Among all models, Steady State Random Waypoint has highest average node degrees ($\bar{D} = 20$) compared to other models given the transmission range is highest (100m). This implies that the movement of devices when following the Steady State Random Waypoint model results in the most well-connected graph compared to the other



(a) $P_{turn} = 0.1$ (b) $P_{turn} = 0.3$

(c) $P_{turn} = 0.5$ (d) $P_{turn} = 0.9$

Figure 9: For the Manhattan grid mobility model, we show how the probability that a path exists varies as a function of transmission range and node speed. We also vary the turn probability from 0.1 to 0.9 for the Manhattan grid mobility model only.

models we consider when the transmission range is 100m. For the original Gauss-Markov model, it has relatively higher connectivity compared to the Manhattan Grid model with a lower turn probability given the transmission range is 100m. Note that in Manhattan grid model, it eventually reaches the same level of connectivity compared to the original Gauss-Markov model when we increase the turn probability to 0.9 given the transmission range is 100m.

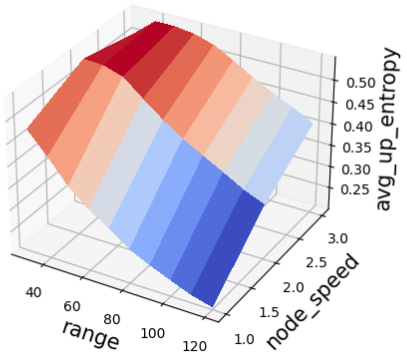
Thus, we see a consistent pattern of how transmission range and node speed affect node degree, despite the variations in the mobility models.

4.2.2 Probability That a Path Exists

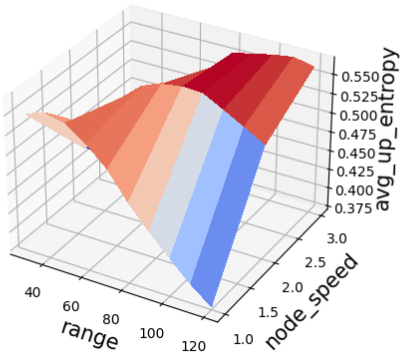
In this section, we examine how our probability that a path exists metric, P_{path} changes as a function of the node transmission range and node speed. Figure 6 (b) shows how P_{path} changes for the Steady-State Random Waypoint Mobility Model, while Figure 7 (b) shows how P_{path} changes for the Original Gauss-Markov Mobility Model, and Figure 9 shows how P_{path} changes for the Manhattan Grid Mobility Model. For all of these models, we vary transmission range and node speed sufficiently to both see the values of P_{path} close to 0 and values of P_{path} close to 1.

For the different mobility models, we see that despite the variations in node mobility, that for all mobility models that as the transmission range increases, the probability that a path exists also increases in the network. We observe that node speed does not seem to have a significant impact on P_{path}, rather that is node density, as set by the transmission range, which determines how likely it is for a path to exist. We note that, while node speed does not seem to affect the probability of a path existing, this does not mean node speed does not affect how long any given path is present in the network.

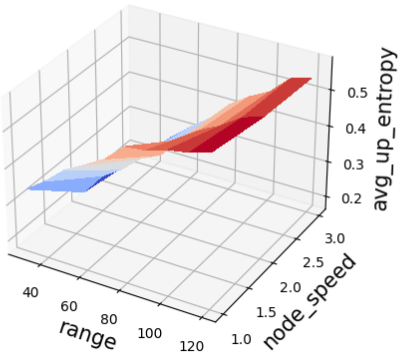
We also note in Figure 9 for the Manhattan Grid Mobility Model, we additionally vary the turn probability at intersections, P_{turn}, from a low probability of turning of $P_{turn} = 0.1$ to a high probability of turning of $P_{turn} = 0.9$. When we increase this turn probability we see that the rate at which the probability of a path exists increases more quickly to 1. I.e., for a smaller transmission range, the probability of a path exists is higher when the turn probability is higher. This is an indication that the increasing the turn probability increases the connectivity of the graph (probably by better distributing nodes in the environment).



(a) Average link-up entropy, Δ=20s



(b) Average link-up entropy, Δ=50s

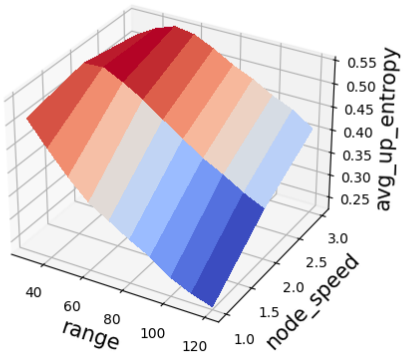


(c) Average link-up entropy, Δ=200s

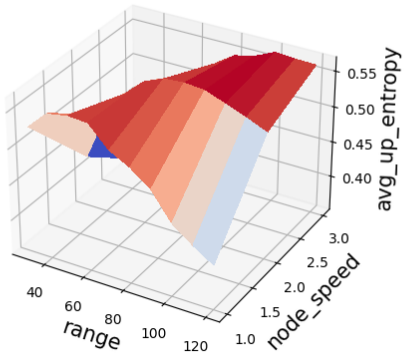
Figure 10: For the steady-state random waypoint mobility model, we show how link-up entropy varies varies as a function of transmission range and node speed with different intervals Δ. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).

4.2.3 Average Link-Up Entropy

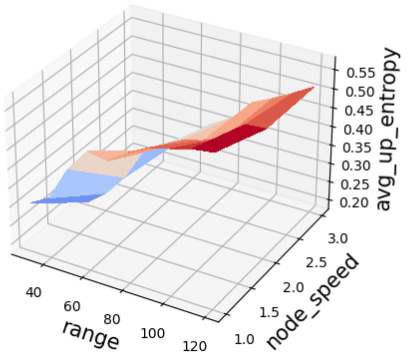
In this section, we examine how the average link-up entropy metric, h_{up} changes as a function of the node transmission range and the node speed. Figure 10 shows how h_{up} changes for the Steady-State Random Waypoint Mobility Model, while Figure 11 shows how h_{up} changes for the Original Gauss-Markov Mobility Model, and Figure 12 shows how h_{up} changes for the Manhattan Grid Mobility Model when the turn probability is set to 0.9. Please see Figures 17 to 12 in Appendix 6.1 to see how h_{up}



(a) Average link-up entropy, Δ=20s



(b) Average link-up entropy, Δ=50s

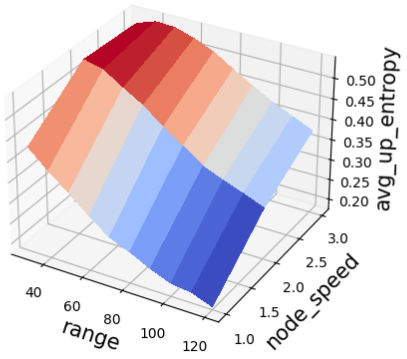


(c) Average link-up entropy, Δ=200s

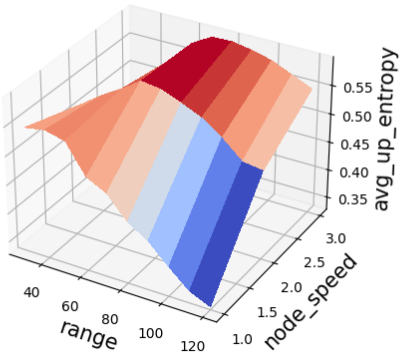
Figure 11: For the original Gauss-Markov mobility model, we show how average link-up entropy varies as a function of transmission range and node speed with different update intervals, Δ. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).

changes for the Manhattan Grid Mobility Model when the turn probability is set to be 0.1, 0.3 and 0.5.

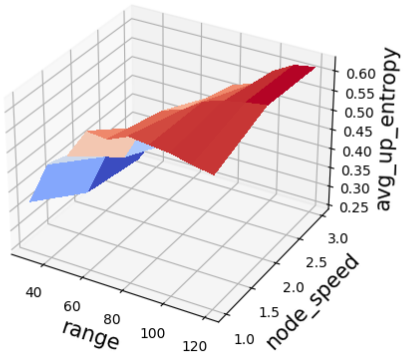
For each of the mobility models we vary the update interval Δ (described in Section 2) from 20, 50, and 200 seconds. We note that more variation is needed in the node speed and transmission range to reach the maximum average link up entropy value of 1: we leave exploring this to future work.



(a) Average link-up entropy, Δ=20s



(b) Average link-up entropy, Δ=50s



(c) Average link-up entropy, Δ=200s

Figure 12: For the Manhattan grid mobility model, we show average link-up entropy as a function of transmission range and node speed when turn probability is 0.9. we also update the interval Δ from 20s to 200s. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).

For the different mobility models, we see that despite the variability in device movement, there are still some similarities. We focus first on what we see when $\Delta = 20$ seconds, which means that we consider only a small period of time in which changes can occur. Recall that the snapshot of the graph we are comparing to is always the first snapshot in the Δ interval. So the shorter the Δ interval, the less likely the

graph at the later timesteps in the interval has changed and how much the graph has changed in this interval depends in part on how predictable the network is. When the node transmission range is large and the node speed is low for $\Delta = 20$ then that is when the network is most well-connected and stable, and we see that is when average link-up entropy is lowest: i.e., the network topology is changing very slowly over time and so the network is very predictable which the link-up entropy reflects. Thus, the link-up entropy across all models is approaching 0 in this situation. When the node transmission range is small and the node speed is high, then that is when the network is least-connected and least predictable within the timeframe of $\Delta = 20$ seconds and that is correspondingly when the average link-up entropy is highest.

We also note that when the node transmission range is large and node speed is high, the network may be well-connected but also not predictable. When the node transmission range is small and the node speed is high, the network may not be well-connected but may still be somewhat predictable. For these scenarios, we need more simulations to fully understand the effects of node speed and transmission range: we leave exploring this for future work.

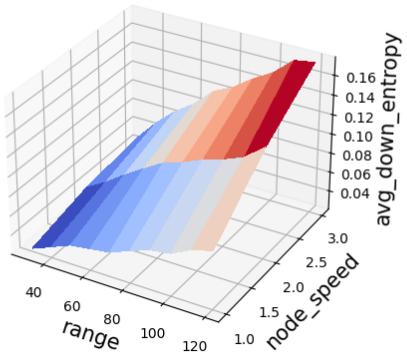
We next focus on the scenarios for when the update interval $\Delta = 50$ seconds. Now we see that when the node transmission range is large and the node speed is low then that is still when the network is most predictable, and correspondingly we see that is when average link-up entropy is lowest. But when node transmission range is small and the node speed is high, due to this high node speed and the larger Δ value, nodes start to predictably see nodes again, even though connections to these nodes are changing. Consequently, we see that average link-up entropy is lower for this setting than the same setting when $\Delta = 20$ seconds. In fact, this setting appears to be the least predictable setting for $\Delta = 20$ as this is correspondingly when the average link-up entropy is lowest.

We next focus on the scenarios for when the update interval $\Delta = 200$ seconds. We

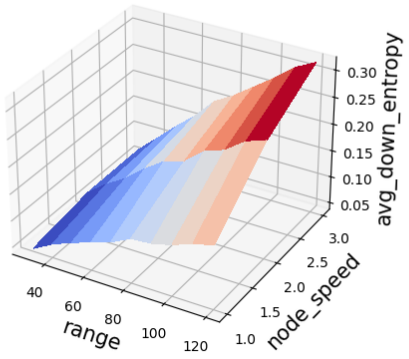
now find that differentiating the impact of node transmission range and node speed on entropy becomes more difficult. We think this is a consequence of two reasons. First, the Δ interval of 200 seconds is a relatively long amount of time to pass in the network, and so there is less connection between the state of the network at the start of the Δ interval and the end of the interval. Second, our simulations are only 900 seconds long. Consequently, there is more noise in the values we compute when $\Delta = 200$ than for the other Δ values since there is less averaging.

Unlike the simulation run for $\Delta = 20$, we propose as future work to run the $\Delta = 200$ simulations for a longer number of time steps to make sure to capture what may be predictable both within a shorter period ($\Delta = 20$) and a longer time period ($\Delta = 200$), but to also capture what may be unpredictable within a short period but is predictable within a long period of time. For instance, in $\Delta = 20$, when the transmission range is low but the node speed is high then that network is unpredictable, but for $\Delta = 200$, when the transmission range is low but the node speed is high then the network becomes predictable, because nodes are regularly reappearing due to their high speed. Consequently, we would suggest using not just one average link-up entropy metric, but several, for different values of Δ, since comparing how entropy changes for these different values of Δ can give insight into the time scale at which node movement is predictable. We propose as future work to run simulations with longer traces and different intervals in the future to understand these issues better.

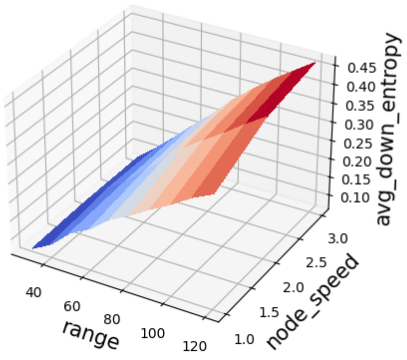
We note that for the simulations run on the Manhattan grid models, varying the turn probability P_{turn} does not seem to have an effect on average link-up entropy. Intuitively, turn probability increases the chance that a node turns at an intersection which will help to distribute the nodes more uniformly in the simulation space. However, according to our simulation results, we do not see a significant impact of turn probability on the average link-up entropy. We will further discuss this in the discussion section.



(a) Average link-down entropy, Δ=20s



(b) Average link-down entropy, Δ=50s



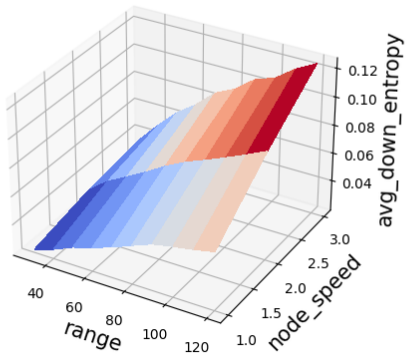
(c) Average link-down entropy, Δ=200s

Figure 13: For the steady-state random waypoint mobility model, we show how link-down entropy varies varies as a function of transmission range and node speed with different intervals Δ. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).

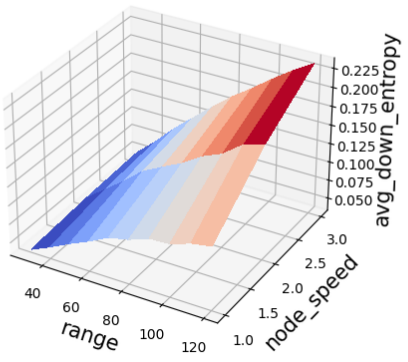
4.2.4 Average Link-Down Entropy

In this section, we examine how the average link-down entropy metric, h_{down} changes as a function of node transmission range and node speed. Figure 13 shows how h_{down} changes for the Steady-State Random Waypoint Mobility Model, while Figure

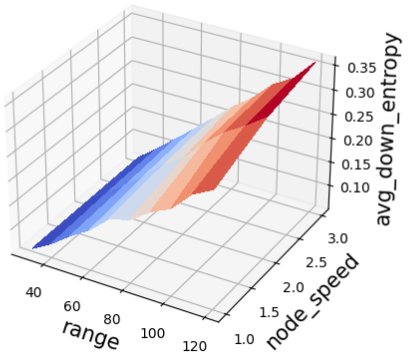
14 shows how h_{down} changes for the Original Gauss-Markov Mobility Model, and Figure 15 for the Manhattan Grid Mobility Model with turn probability set to 0.9. Please see Figures 19 to 21 in Appendix 6.2 for the average link-down entropy graphs for when the turn probability is set to be 0.1, 0.3 and 0.5 for the Manhattan Grid Mobility Model.



(a) Average link-down entropy, Δ=20s



(b) Average link-down entropy, Δ=50s



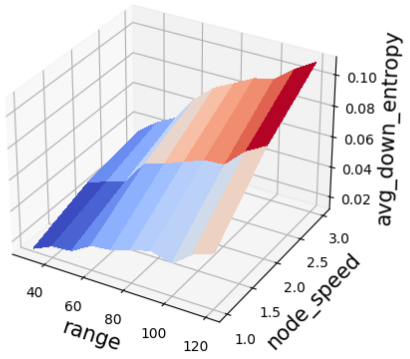
(c) Average link-down entropy, Δ=200s

Figure 14: For the original Gauss-Markov mobility model, we show how average link-down entropy varies as a function of transmission range and node speed with different intervals Δ. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c)

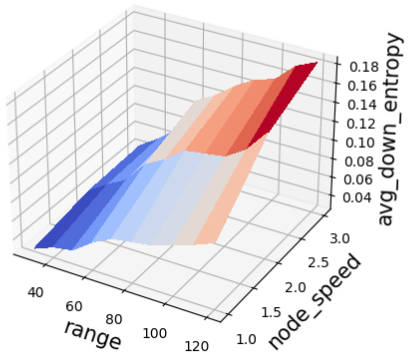
We observe that for all of the mobility models, despite the variation in mobility and regardless of the update interval Δ, the largest average link-down entropy values are when both the node transmission range is large and the node speed is high. That is, this is the setting for when the links transitioning from down to up are most unpredictable. We also observe that that as the update interval Δ increases from 20 seconds to 200 seconds the maximum value for average link-down entropy also increases, from about 0.10 to 0.30. This indicates that links appearing is also becoming increasingly unpredictable.

For all of the mobility models, we also observe that when the node transmission range is small and the node speed is small, that the average link-down entropy is small, indicating that the scenario is quite predictable. That is, few links are appearing regardless of Δ.

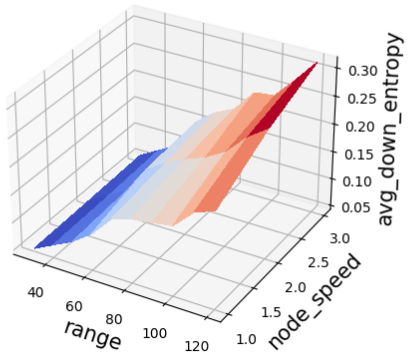
Finally, we note that while there are similarities between the average link-down entropy plots and the node degree plots, the node degree plots do not vary significantly as a function of node speed, unlike the average link-down entropy plots.



(a) Average link-down entropy, Δ=20s



(b) Average link-down entropy, Δ=50s



(c) Average link-down entropy, Δ=200s

Figure 15: For the Manhattan grid mobility model, we show average link-down entropy as a function of transmission range and node speed when turn probability is 0.9. We also update the interval Δ from 20s to 200s. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).

4.3 Discussion

One of the issues we had in our simulations was the time it took to run. This is because we treat every second in a 900 second simulation as a separate graph to analyze. Consequently, we had less simulation time for averaging our results,

leading us to not be able to fully explore and understand certain network settings, particularly as to how they impacted average link-up entropy and average link-down entropy.

For a given network or node within a network, knowing something about the values of the four metrics, node degree, probability path exists, average link-up entropy, and average link-down entropy can be helpful in guiding routing. For instance, when the network has low predictability (in terms of the entropy measures) and low connectivity (in terms of the node degree and probability path exists metrics), then this is when it is likely better to use a flooding based forwarding strategy rather than collecting routing state and sending a single packet copy.

We observe that regardless of mobility model, that by varying the node transmission range and node speed we can see similar values for our different metrics. This tells us that there it should be possible to transform different mobility models and kinds of device mobility to a common feature space based on our metrics.

Combining the metrics may additionally give us ways to distinguish the important differences of the mobility models. For instance, while the average link-up entropy may have similar trends across the different mobility models, the use of node degree can be used to help distinguish the slightly more well-connected random waypoint mobility model from the less connected models.

For the turn probability variation in the Manhattan grid model, our simulations are not fully developed since the average link-up entropy never reaches one in our simulations (that is, the maximum average entropy in the simulation is around 0.5). We might see differences when we have a higher average link-up entropy. Further experiments with additional values of transmission range and node speed are needed to find out if the turn probability has an effect on the average link-up and link-down entropies.

4.4 Possible Future Experiments

In this section, we describe some possible experiments to be done in the future, varying different settings. In our simulations, we vary a number of different network settings including transmission range, turn probability and node speed to indirectly vary the connectivity and predictability of the network scenarios we consider. There are two additional network settings that we believe are interesting to vary.

4.4.1 Velocity update frequency ($-q$) for original Gauss-Markov model

Velocity update frequency is used in the original Gauss-Markov model to determine how node speed changes in each simulation. Currently, we are using a node update of 30 in the original Gauss-Markov model simulations. It will be interesting to see how predictability of the network changes over time with different update frequencies. One way to vary the possible ranges for the update frequency is $[30, 50, 100]$.

4.4.2 Weighting ($-\alpha$) for original Gauss-Markov model

This weighting determines how the new direction in which nodes move is affected by their previous direction. This network setting is also worth altering because in the Manhattan grid model, we do see that an increase in turn probability increases the average node degree and more quickly increases the probability that a path exists gets to 1 for a relatively lower transmission range. Therefore, changes in how node directions are updated might have some impact on the connectivity and predictability of a given mobile ad hoc network. One set of the potential ranges that we plan to use for α is $[0.1, 0.3, 0.7]$.

4.4.3 Additional Simulations on longer time steps [$d = 10000$]

We can further improve our simulations and provide more insight into network dynamics in mobile ad hoc networks by increasing the simulation duration of the traces we generate. Due to the computational limits when computing our metrics, our current simulations only run for 900 time steps. We plan to increase the trace generation to a longer time period (10000 seconds) and use cloud computing so that we can better sample the parameters we vary, and have more samples when computing the time-averaged values of the metrics we propose. Additionally, generating longer traces will allow us to investigate larger Δ update intervals when we are calculating the values of the entropy-based metrics. It will be particularly interesting to see how network predictability changes as a function of larger values of Δ.

4.4.4 Machine Learning Applications

We expect that the metrics we have proposed could be used as features into a machine learning model to make predictions about other useful metrics for problems in computer network as well as applications of dynamic graphs more generally.

5 Conclusions

This goal of this thesis was to explore ways to characterize device mobility in a MANET. We propose four metrics to quantify network connectivity and predictability over time, specifically average node degree, probability that a path exists, average link-up entropy, and average link-down entropy. Our results show that these metrics, when computed for different mobility models, show common trends as a function of node transmission range and node speed. In future work, we propose exploring how to use these as metrics for making routing decisions when the network mobility is

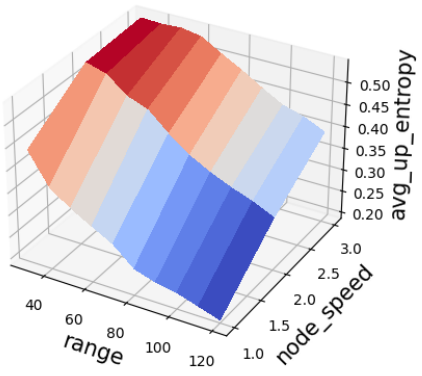
unknown.

6 Appendix

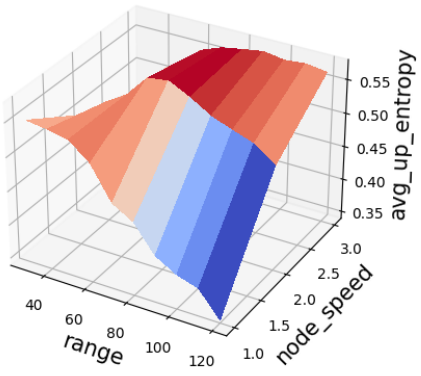
In this section, we include additional graphs for the Manhattan grid mobility model, varying the path turn probability, and show results for how the average-link-up entropy and average link-down entropy varies.

6.1 Average Link-up Entropy

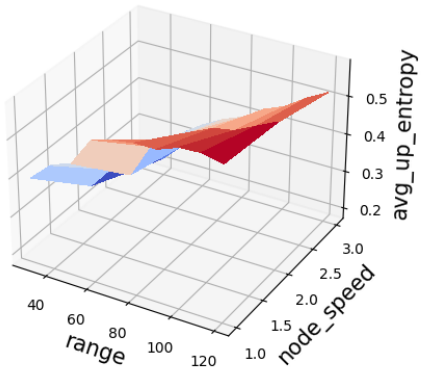
In this section, we show how the average link-up entropy varies for the Manhattan grid mobility model as the turn probability is varied.



(a) Average link-up entropy, Δ=20s

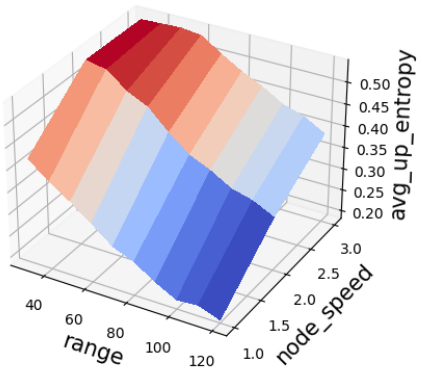


(b) Average link-up entropy, Δ=50s

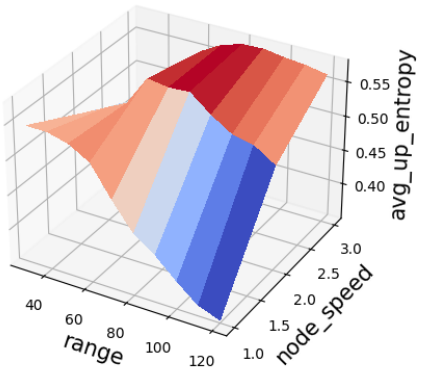


(c) Average link-up entropy, Δ=200s

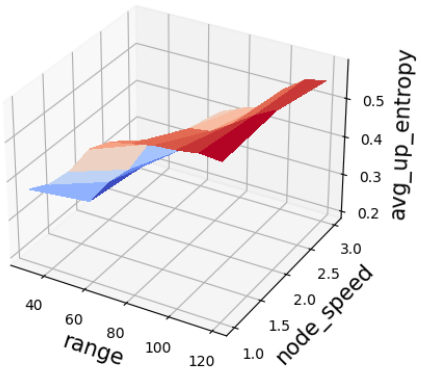
Figure 17: For the Manhattan grid mobility model, we show average link-up entropy as a function of transmission range and node speed when turn probability is 0.3. we also update the interval Δ from 20s to 200s. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).



(a) Average link-up entropy, Δ=20s



(b) Average link-up entropy, Δ=50s

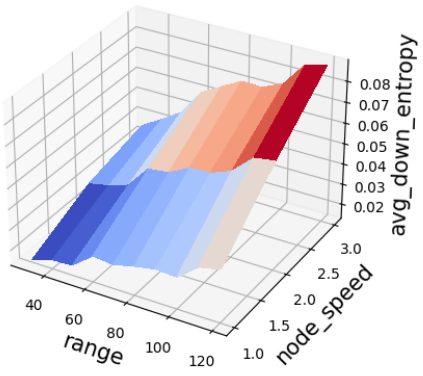


(c) Average link-up entropy, Δ=200s

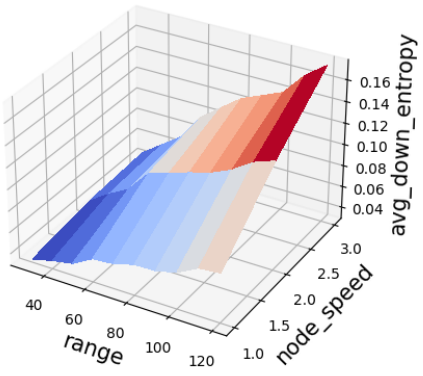
Figure 18: For the Manhattan grid mobility model, we show average link-up entropy as a function of transmission range and node speed when turn probability is 0.5. we also update the interval Δ from 20s to 200s. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).

6.2 Average Link-down Entropy

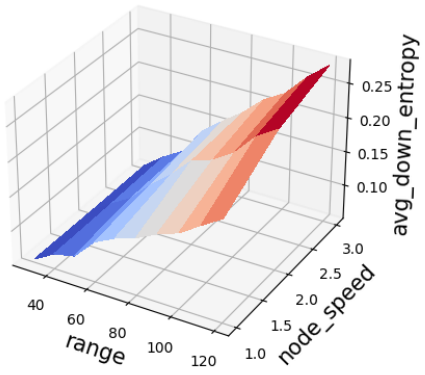
In this section, we show how the average link-down entropy varies for the Manhattan grid mobility model as the turn probability is varied.



(a) Average link-down entropy, Δ=20s

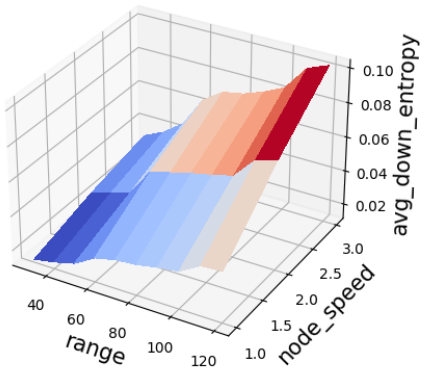


(b) Average link-down entropy, Δ=50s

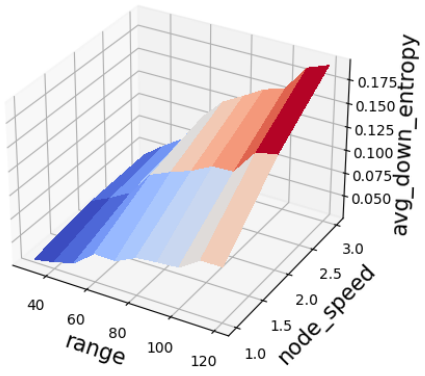


(c) Average link-down entropy, Δ=200s

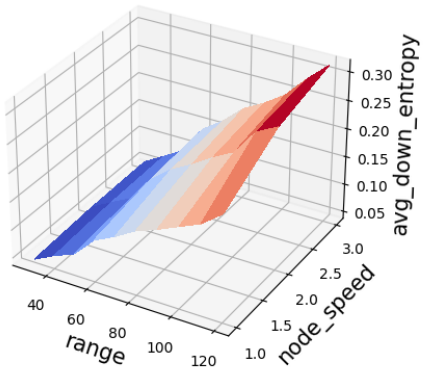
Figure 19: For the Manhattan grid mobility model, we show average link-down entropy as a function of transmission range and node speed when turn probability is 0.1. we also update the interval Δ from 20s to 200s. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c)



(a) Average link-down entropy, Δ=20s

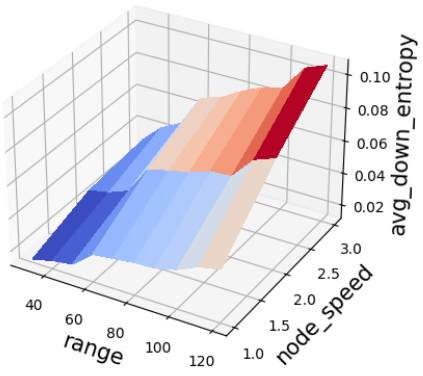


(b) Average link-down entropy, Δ=50s

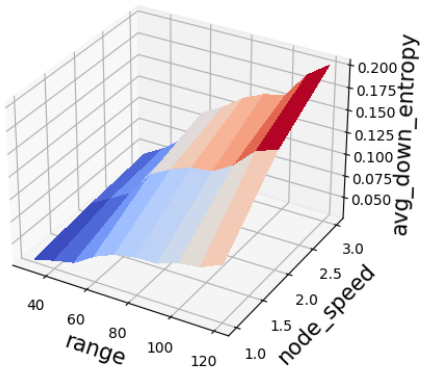


(c) Average link-down entropy, Δ=200s

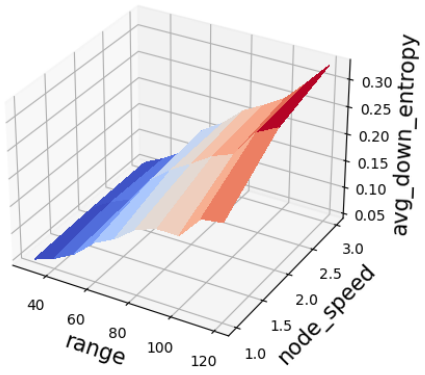
Figure 20: For the Manhattan grid mobility model, we show average link-down entropy as a function of transmission range and node speed when turn probability is 0.3. we also update the update interval Δ from 20s to 200s. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).



(a) Average link-down entropy, Δ=20s

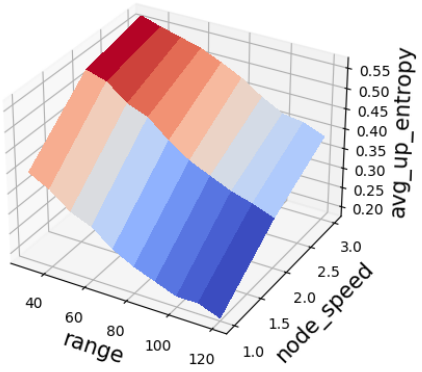


(b) Average link-down entropy, Δ=50s

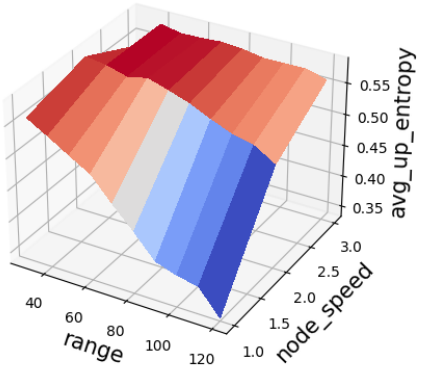


(c) Average link-down entropy, Δ=200s

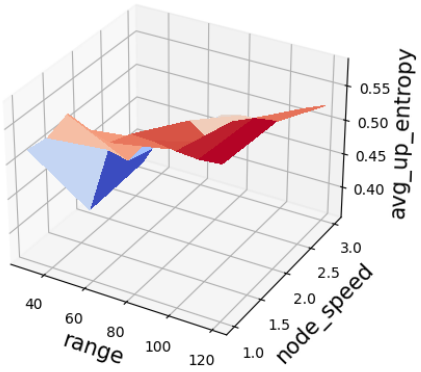
Figure 21: For the Manhattan grid mobility model, we show average link-down entropy as a function of transmission range and node speed when turn probability is 0.5. we also update the update interval Δ from 20s to 200s. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).



(a) Average link-up entropy, Δ=20s



(b) Average link-up entropy, Δ=50s



(c) Average link-up entropy, Δ=200s

Figure 16: For the Manhattan grid mobility model, we show average link-up entropy as a function of transmission range and node speed when turn probability is 0.1. we also update the interval Δ from 20s to 200s. We set $\Delta = 20$s for (a), $\Delta = 50$s for (b), and $\Delta = 200$s for (c).

References

[1] M. Abolhasan, T. Wysocki, and E. Dutkiewicz. A review of routing protocols for mobile ad hoc networks. *Ad Hoc Networks*, 2(1):1–22, 2004.

[2] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn. Bonnmotion: a mobility scenario generation and analysis tool. In *3rd International ICST Conference on Simulation Tools and Techniques*, 2010.

[3] Y. Bi, H. Shan, X. S. Shen, N. Wang, and H. Zhao. A multi-hop broadcast protocol for emergency message dissemination in urban vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 17(3):736–750, 2015.

[4] I. Chlamtac, M. Conti, and J. J.-N. Liu. Mobile ad hoc networking: imperatives and challenges. *Ad Hoc Networks*, 1(1):13–64, 2003.

[5] E. T. S. I. (ETSI). Universal mobile telecommunicatios system (umts) - selection procedures for the choice of radio transmission technologies of the umts, umts 30.03 version 3.2.0, tr 101 112 ed. 1998.

[6] L. Jianmin, W. Qi, H. Chentao, and X. Yongjun. Ardeep: Adaptive and reliable routing protocol for mobile robotic networks with deep reinforcement learning. In *2020 IEEE 45th Conference on Local Computer Networks (LCN)*, pages 465–468. IEEE, 2020.

[7] S. Kaviani, B. Ryu, E. Ahmed, K. Larson, A. Le, A. Yahja, and J. H. Kim. Deepcq+: Robust and scalable routing with multi-agent deep reinforcement learning for highly dynamic networks. In *MILCOM 2021-2021 IEEE Military Communications Conference (MILCOM)*, pages 31–36. IEEE, 2021.

[8] M. Kwon, J. Lee, and H. Park. Intelligent IoT connectivity: Deep reinforcement learning approach. *IEEE Sensors Journal*, 20(5):2782–2791, 2019.

[9] B. Liang and H. Zygmunt. Predictive distance-based mobility management for multidimensional pcs networks. *IEEE/ACM Trans. Netw.*, 11:718–732, 10 2003.

[10] V. Manfredi, M. Crovella, and J. Kurose. Understanding stateful vs stateless communication strategies for ad hoc networks. In *Proceedings of the 17th annual international conference on Mobile computing and networking*, pages 313–324, 2011.

[11] W. Navidi and T. Camp. Stationary distributions for the random waypoint mobility model. *IEEE Transactions on Mobile Computing*, 3(1):99–108, 2004.

[12] K. Poularakis, Q. Qin, E. M. Nahum, M. Rio, and L. Tassiulas. Flexible SDN control in tactical ad hoc networks. *Ad Hoc Networks*, 85:71–80, 2019.