Simulation Configuration and Performance Metrics EZBRP, SACBRP, BIABRP

#1Senthil Jayapal, #2Annadurai Manickam nadar, #3Ramesh Palanisamy
Department of Information Technology,
University of Technology and Applied Sciences- IBRA
Sultanate of Oman
1jayapal@ict.edu.om, 2annadurai@ict.edu.om, 3palanisamy@ict.edu.om

Abstract-Network Simulator (Version 2), widely known as NS2, is simply an event-driven simulation tool that has proved useful in studying the dynamic nature of communication networks. Both wired and wireless network functions and protocols (e.g., routing algorithms, TCP, UDP) can be simulated using NS2. In general, NS2 provides users with a method of specifying such network protocols and simulating their corresponding behaviors.

1. INTRODUCTION

Ever since its birth in 1989, NS2 has gained constant popularity in the networking research community due to its flexibility and modular nature. Several revolutions and revisions have marked the growing maturity of the tool, thanks to the some of the key players in this field. Among these are the University of California and Cornell University who developed the REAL network simulator 1, the foundation of NS is on. Since 1995 the Defense Advanced Research Projects Agency (DARPA) has been supporting the development of NS through the Virtual InterNetwork Testbed (VINT) project. Currently, the National Science Foundation (NSF) has joined the ride in development. In addition to all these, a group of researchers and developers in the community are constantly working to continue to ensure that NS2 strong and versatile.

2. ARCHITECTURE OF NS2

Fig 6.1, shows the basic architecture of NS2. NS2 provides users with an executable command ns which takes on input argument, the name of a Tcl simulation scripting file. Users feed the name of a Tcl simulation script (which sets up a simulation) as an input argument of an NS2 executable command ns. In most cases, a simulation trace file is created, and is used to plot graph and/or to create animation. NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (i.e., a backend) of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects and scheduling discrete events (i.e., a frontend). The C++ and the OTcl are linked together using TclCL. Mapped to a C++ object, variables in the OTcl domains are sometimes referred to as handles. Conceptually, a handle (e.g., n as a Node handle) is just a string in the OTcl domain, and does not contain any functionality. Instead, the functionality (e.g., receiving a packet) is defined in the mapped C++

object (e.g., of class Connector). In the OTcl domain, a handle acts as a frontend which interacts with users and other OTcl objects. It may define its own procedures and variables to facilitate the interaction. The member procedures and variables in the OTcl domain are called instance procedures (instprocs) and instance variables (instvars), respectively. It is important to the readers are to learn about C++ and OTcl languages to get a better understanding of these architecture.

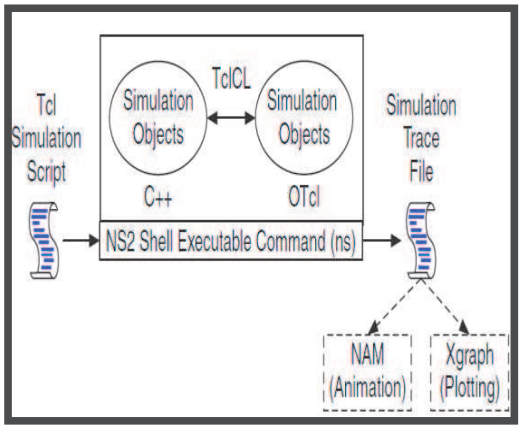

Fig 6.1: Basic architecture of NS.

NS2 provides a large number of built-in C++ objects. It is advisable to use these C++ objects to set up a simulation using a Tcl simulation script. However, advance users may find these objects to be insufficient. They need to develop their own C++ objects, and use a OTcl configuration interface to put together these objects. After simulation, NS2 outputs either text-based or animation-based simulation results. To interpret these results both graphically and interactively, tools such as NAM (Network AniMator) and XGraph are used. To analyze a particular behavior of the network, users can extract a relevant subset of text-based data and transform it to a more conceivable presentation.

3. DISCRETE-EVENT SIMULATION

NS2 is a discrete-event simulator, where actions are associated with events rather than time. An event in a discrete-event simulator consists of execution time, a set of actions, and a reference to the next event (Fig 6.2). These events connect to each other and form a chain of events on the simulation timeline. Unlike a time-driven simulator, in an event-driven simulator, the time between a pair of events does not need to be constant. When the simulation starts, events in the chain are executed from left to right (i.e. chronologically). The next section, discusses the simulation concept of NS2.

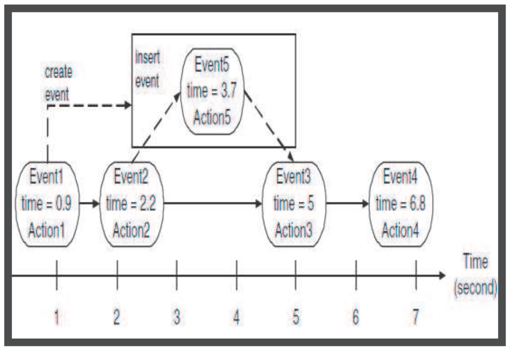
create event
insert event
Event5 time = 3.7 Action5
Event1 time = 0.9 Action1
Event2 time = 2.2 Action2
Event3 time = 5 Action3
Event4 time = 6.8 Action4
Time (second)
1
2
3
4
5
6
7

Fig 6.2: Discrete-Event Simulation.

Fig 6.2 demonstrates a sample chain of events in a discrete-event simulation. It can be observed that event contains execution time and a reference to the next event. In this figure, Event1 creates and inserts Event5 after Event2 (the execution time of Event 5 is at 3.7 second).

i. NS2 Simulation Concept

NS2 simulation consists of two major phases.

Phase I: Network Configuration Phase

In this phase, NS2 constructs a network and sets up an initial chain of events. The initial chain of events consists of events which are scheduled to occur at certain times (e.g., start FTP (File Transfer Protocol) traffic at 1 second.). These events are called at-events. This phase corresponds to every line in a Tcl simulation script before executing instproc run{} of the Simulator object.

Phase II: Simulation Phase

This part corresponds to a single line, which invokes instproc Simulator::run {}. Ironically, this single line contributes to most of the simulation (e.g., 99%). In this part, NS2 moves along the chain of events and executes each event chronologically. Here, the instproc Simulator::run{} starts the simulation by dispatching the first event in the chain of events. In NS2, "dispatching an event" or "firing an event" means "taking actions corresponding to that event". An action, for example, refers to starting FTP traffic or creating another event and inserting the created event into the chain of events. In Fig 6.2, at 0.9 s, Event1 creates Event5, which will be dispatched at 3.7 s, and inserts Event5 after Event2. After dispatching an event, NS2 moves down the chain and dispatches the next event. This process repeats until the last event corresponding to instproc halt{} of OTcl class Simulator is dispatched, signifying the end of simulation.

4. NS2 COMPONENTS

A network object is one of the main NS2 components, which is responsible for packet forwarding. NS2 implements network objects using the polymorphism concept in Object-Oriented Programming (OOP). Polymorphism allows network objects to take different actions ways under different contexts. For example, a Connector object immediately passes the received packet to the next network object, while a Queue1 object enques the received packets and forwards only the head of the line packet.

Based on the functionality, NS2 modules (or objects) can be classified into following four types:

- ✓ **Network objects** are responsible for sending, receiving, creating, and destroying packet-related objects. Since these objects are those derived from class NsObject, hereforth, they will be referred to as NsObjects.
- ✓ **Packet-related objects** are various types of packets which are passed around a network.
- ✓ **Simulation-related objects** control simulation timing, and supervise the entire simulation. Some examples of simulation-related objects are events, handlers, the Scheduler, and the Simulator.
- ✓ **Helper objects** do not explicitly participate in packet forwarding. However, they implicitly help to complete the simulation. For example, a routing module calculates routes from a source to a destination, while network address identifies each of the network objects.

5. SIMULATION OF CRAHNS USING NETWORK SIMULATOR 2

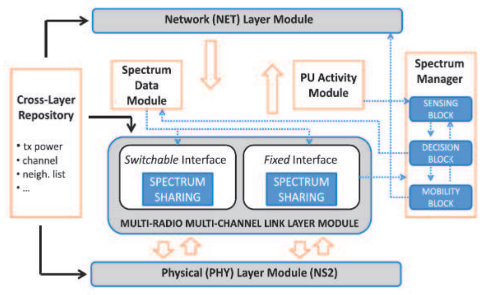

Fig. 6.4. NS2-CRAHN architecture

It has been extended the NS-2 simulator with various blocks to be able to evaluate the performance of various approaches for CRAHNs. Fig. 6.4 shows the architectural model of the NS-2 CRAHN simulator. Compared to the traditional NS-2 architecture, it have added the following features, implemented as extendible stand-alone C++ modules:

- ✓ **PU Activity module:** It describes the characteristics of active PUs in the current scenario, including operating channel, physical location, and transmitting range. It also contains the description of PU activity in each spectrum band, as a sequence of ON and OFF periods over simulation time. All the information about PUs are contained in a PU-log file, which is generated offline. The format of PU-log file is described in Section 6.6.1.
- ✓ **Spectrum data module:** It describes the PHY characteristics of each channel, such as operating frequency, channel capacity and average bit error rate (BER). The format of channel-log file is described in Section 6.6.2.
- ✓ **Spectrum manager module:** It implements the cognitive cycle for each CR user, as described in Section 6.6.3. It is composed of three main blocks: the spectrum sensing block, the spectrum decision block and the spectrum mobility block. The spectrum sensing block is responsible for detecting the activity of PUs on the current channel. To this aim, the spectrum sensing block interacts with the PU activity module. In the case of PU detection, the spectrum decision block chooses the policy to be adopted, i.e. whether to switch to a new channel or to stay on the current channel. If a channel switch is required, the spectrum decision block can choose the next available channel to be used for CR user operation, and the spectrum mobility block manages the spectrum handoff process.
- ✓ **Multi-radio multi-channel link layer module:** It implements the multi-radio multi-channel environment for each CR user. Section 6.6.4 provides the details of the link-layer coordination, among the available radio interfaces. Each radio implements the spectrum sharing block for distributed channel access in wireless networks. Current implementation is based on the CSMA/CA MAC scheme. The spectrum sharing block interacts with the PU activity module to model the interference caused by PUs on current ongoing transmissions of CR users.
- ✓ **Network layer module:** Traditional routing protocols for wireless ad hoc networks can be used at network layer, as well as customized network protocols for CRAHNs.

- ✓ **Cross-layer repository module:** It enables information sharing among protocols at different layers of the protocol stack. For example, it may contain information collected at the PHY layer (e.g. current transmitting power), MAC layer (e.g. current size of the backoff window) and network layer (e.g. current neighbors' list).

Following sections provide a detailed description of the PU-log (Section 6.6.1) and channel-log file (Section 6.6.2), of the spectrum manager functionalities (Section 6.6.3) and of the rationale of the CR user model (Section 6.6.4).

6. CHANNEL-LOG FILE FORMAT

The channel-log file contains information about

(i) physical channel characteristics and

(ii) channel quality.

The spectrum data module is responsible for loading the information from the file and making them available to the other modules. For each CR channel, the log file contains an entry with this format

$$(id, frequency, bandwidth, noise) \;---\; (3)$$

where id is the identifier of the channel (a number between 1 and N), frequency is the channel central frequency, bandwidth is the raw bandwidth of the channel (e.g. 11 Mb/s), and noise is the average value of the noise on that channel. By using bandwidth and noise, and by knowing the power received at a given location, it is possible to model the average BER (Bit Error Rate) experienced by the receiver node, for the QPSK modulation.

7. SPECTRUM MANAGER

The spectrum manager module implements the cognitive cycle for each CR user, by using the spectrum sensing block, the spectrum decision block and the spectrum mobility block.

It emphasize here that additional spectrum policies and spectrum allocation algorithms can be easily integrated into the current module, by considering metrics provided by the MAC, physical or routing layer, or any combination among them. This also facilitates mechanisms that allow a CR user to temporarily route around the PU activity on a different frequency portion until the current PU activity stops.

Spectrum Mobility Block

The spectrum mobility block is invoked when the spectrum decision block decides that the CR user must vacate the current channel. It receives from the spectrum decision block the new channel to switch to (e.g. next_channel). The delay induced by the channel switch is modeled by using a timer. A CR user is not allowed to utilize the radio interface for communication during the handoff operation. When the handoff process is completed, the spectrum sensing block is invoked to detect the PU activity on next_channel. If next_channel is found free of PU activities, then a spectrum handoff notification is sent to the upper layer, and protocol reconfiguration is performed at the network layer.

8. CR USER MODEL

The CR user model is shown in Fig. 6.5.

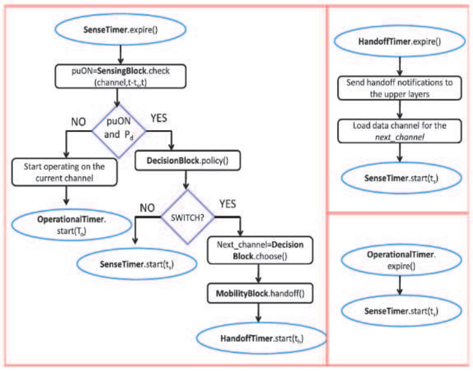
SenseTimer.expire()
puON=SensingBlock.check (channel,t-t_s,t)
puON and P_d
NO
YES
Start operating on the current channel
DecisionBlock.policy()
OperationalTimer.start(T_o)
SWITCH?
NO
YES
SenseTimer.start(t_s)
Next_channel=DecisionBlock.choose()
MobilityBlock.handoff()
HandoffTimer.start(t_h)
HandoffTimer.expire()
Send handoff notifications to the upper layers
Load data channel for the next_channel
SenseTimer.start(t_s)
OperationalTimer.expire()
SenseTimer.start(t_s)

Fig. 6.5. The CR model implemented in the NS2-CRAHN simulator.

Link Layer Management

Each CR user at initialization designates one interface as its fixed interface, and the second interface as the switchable interface. Moreover, each CR user periodically informs its neighbors about the channel used by its fixed interface, by broadcasting an HELLO message on all the available channels. When a CR user (e.g. node A) needs to communicate with a neighbor node (e.g. node B), it tunes its switching interface to the channel used by the fixed interface of node B, and starts transmitting.

Each CR user performs a sensing cycle on the fixed radio interface, by periodically switching between two states: a sensing state (for a ts time interval) and operational state (for a To time interval). To this aim, each CR user (say node C) is associated with a sensing timer and an operational timer. When the sensing timer expires, C stops sending/receiving data on the current channel and performs sensing operations to detect the presence of a PU. In the case of channel switching to next_channel, a broadcast message is sent by node C to inform its neighborhood about the channel used by its fixed interface.

Spectrum Sharing

Each interface implements a spectrum sharing scheme, based on a carrier sensing multiple access with collision avoidance (CSMA-CA) MAC scheme, with acknowledgments (ACK) and frame retransmissions at the MAC layer. It extend the MAC scheme to take into account the interference caused by PUs on CR users.

9. MAIN COMPONENTS OF A SIMULATION

Interpreted Hierarchy

Created by various instprocs, the main OTcl simulation components are as follows:

- ✓ **The Scheduler** (scheduler_ created by instproc Simulator::init)maintains the chain of events and executes the events chronologically.
- ✓ **The null agent** (nullAgent_ created by instproc Simulator::init) provides the common packet dropping point.5
- ✓ **Node reference** (Node_ created by instproc Simulator::node) is an associative array whose elements are the created nodes and indices are node IDs.
- ✓ **Link reference** (link_ created by instprocs simplex-link{...} or duplexlink{...}) is an associative array. Associated with an index with format "sid:did", each element of link_ is the created uni-directional link which carries packet from node "sid" to node "did".

a. SETTINGS AND PERFORMANCE METRICS (Need to be corected at last)

The default simulation settings are as follows: We simulate an area of $4{,}000 \times 3{,}000\ m^2$ with 6 PUs and 1,334 SUs. The transmission range of PUs and SUs is 250 m. We run the simulation 8,000 times, and the PUs and SUs are uniformly distributed at random with a different random seed in each simulation run. We fix 30 pairs of source and destination PUs with random relative locations in each simulation run. SACBRP is used only when the source and destination SUs are both outside of any PU region; otherwise, TIGHT is used to send packets over the secondary channel only. Each source SU sends one packet of 512 bytes to its destination SU in each simulation run. Therefore, totally 8,000 packets are sent between each source destination SU pair, and each data point in subsequent figures represents the average for 240 thousand packets (unless stated otherwise).

b. MOBILITY MODEL

An important factor in mobile ad-hoc networks is the movement of nodes, which is characterized by speed, direction and rate of change. Mobility in the “physical world” is unpredictable, often unrepeatable, and it has a dramatic effect on the protocols developed to support node movement. Therefore, different “synthetic” types of mobility models have been proposed to simulate new protocols. The term ‘Synthetic’ means to realistically represent node movement without using network traces.

Camp et al., 2002 discusses the following seven different synthetic entity mobility models based on random directions and speeds:

1. **Random Walk Mobility Model:** A simple mobility model based on random directions and speeds.
2. **Random Waypoint Mobility Model:** A model based on random waypoints and random speeds that includes pause times between changes in destination and speed, which will be discussed in more detail in the following section.
3. **Random Direction Mobility Model:** A model that forces mobile nodes to travel to the edge of the simulation area before changing direction and speed.
4. **Boundless Simulation Area Mobility Model:** A model that converts a 2D rectangular simulation area into a torus-shaped simulation area.
5. **Gauss-Markov Mobility Model:** A model that uses one tuning parameter to vary the degree of randomness in the mobility pattern.
6. **Probabilistic Version of the Random Walk Mobility Model:** A model that utilizes a set of probabilities to determine the next position of a mobile node.
7. **City Section Mobility Model:** A simulation area that represents streets within a city.

To thoroughly and systematically study a new Ad-hoc Network protocol, it is important to simulate this protocol and evaluate its protocol performance. Protocol simulation has several key parameters, including mobility model and communicating traffic pattern, among others.

i.Random-Based Mobility Models

In random-based mobility models, the mobile nodes move randomly and freely without restrictions. To be more specific, the destination, speed and direction are all chosen randomly and independently of other nodes. This kind of model has been used in many simulation studies.

ii. Random Waypoint Mobility Model

The random waypoint (RWP) mobility model has been widely used in mobile ad-hoc network simulations. This mobility model is a simple and straightforward stochastic model. In RWP, a mobile node moves on a finite continuous plane from its current position to a new location by randomly choosing its destination coordinates, its speed of movement, and the amount of time it will pause before when it reaches the destination. On reaching the destination, the node pauses for some time distributed according to some random variable and the process repeats itself.

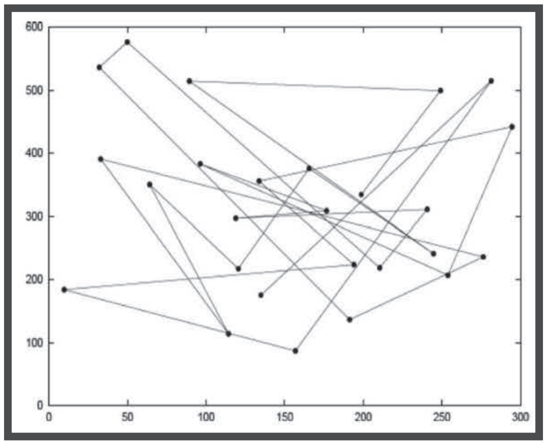

Fig 6.6 Traveling pattern of mobile node using the Random Waypoint Mobility model

iii. Transition Length and Duration

The RWP model correlates the speed and the direction change behavior. The time between two direction change events is no longer an adjustable input parameter of the model, later it depends on the speed of the nodes and the size and shape of the area. For a given area, a higher speed results in a higher frequency of direction changes. The speed behavior and the direction change behavior are investigated.

Consider RWP movement in a rectangular area of size *a* × *b* and again derive the distribution of the transition length *L*. Without loss of generality it is assumed that *a* ≥ *b*. The spatial distribution of the 2D waypoints *P* = (*Px*, *Py*) is now given by the uniform distribution

$$f P_x P_y(x,y) = \begin{cases} \frac{1}{ab} for\ 0 \leq x \leq a\ and\ 0 \leq y \leq b \\ 0\ \ else \end{cases} \quad ---(1)$$

The distance between two points *P*1 = (*Px*$_1$, *Py*$_1$) and *P*2 = (*Px*$_2$, *Py*$_2$) is

$$L = \|P_2 - P_1\| = \sqrt{|P_{x1} - P_{x2}|^2 + |P_{y1} - P_{y2}|^2}$$

$$= \sqrt{L_x^2 + L_y^2} \quad ---(2)$$

Note that the random variable $L_x = |P_{x1} - P_{x2}|$ represents the random distance between two uniformly distributed coordinates *Px*1 and *Px*2 on a 1D line segment [0, *a*]. The same holds true for $L_y = |P_{y1} - P_{y2}|$ if it is replaced *a* by *b*. In addition, both random distances are independent of each other, and therefore the joint *pdf* of L_x and L_y is given by

$$fL_{x,}L_y(l_x, l_y) = fL\,(l_x)fL\,(l_y) = \frac{4}{a^2b^2}\,(-l_x + a)\,(-l_y + b)\text{for } 0 \leq l_x \leq a \text{ and } 0 \leq l_y \leq b \text{ and } 0, \text{otherwise} ---(3)$$

With this expression, it is possible to derive the cdf Pr $(L \leq l)$ by integration of L_x,L_y $(l_{x,}l_y)$ over the circular area$D = \sqrt{l_x^2 + l_y^2} \leq l$ in the *(lx-ly)*- space, that is,

$$\Pr(L \leq l) = \iint_D^d f_{L_x} f_y\ (l_x, l_y) ---(4)$$

As in the case of 1D, it cannot compute this integral in a straightforward manner, namely, by setting the right-hand side of Equation (3) in Equation (4), but must take into account that $fL_x, L_y\,(l_x, l_y) = 0\, for\ l_x > a\ or\ l_y > a$. Thus , it is distinguished between three cases:

Solving these integrals, taking the derivative with respect to *l*, and performing some trigonometric simplifications lead to the *pdf* of the transition length *L* of nodes moving according to the RWP model in a rectangular area of size *a* × *b*, *a* ≥ *b*:

$$f_L(l) = \frac{4l}{a^2b^2} f_0(l) ---(6)$$

With

For arbitrary a, the value of $f_L(l)$ is obtained by $f_L(l) = \frac{1}{a} f_l$(l). The expected value of L is

$$E\{L^2\} = \frac{1}{15}\left[\frac{a^2}{b^2} + \frac{b^2}{a^2} + \sqrt{a^2+b^2}\,(3 - \frac{a^2}{b^2} - \frac{b^2}{a^2})\right] + \frac{1}{6}\left[\frac{b^2}{a} \operatorname{arccos} h\ \frac{\sqrt{a^2+b^2}}{b} + \frac{a^2}{b} \operatorname{arccos} h\ \frac{\sqrt{a^2+b^2}}{a}\right] ---(8)$$

With arcos h (x) = In $(x + \sqrt{x^2 - 1})$. Figure 4.4 shows the curve for $\mathrm{E}\{L\}$/a over (b/a). For example, the expected length within a square a size a x a is $\mathrm{E}\{L\}$ = The second moment of L is given by $\mathrm{E}\{L^2\} = \frac{1}{6}\,(a^2 + b^2)$

It should be noted that the moments for the ID case are obtained, that is, $lim_{b\to\infty}\ E\{L\} = \frac{1}{3}a\ and\ limb_{b\to 0}\ E\{L\} = \frac{1}{6}a^2$.

The distance *pdf* $f_L(l)$ of a circular system area of radius a is derived as follows: i. P=p is the starting waypoint, ii. The conditional probability $P_r(L \leq l/p = p)$ is derived using basic geometric equation on the intersection area of two circles in polar coordinates, and iii. P$(L \leq L)$is the integration of the *pdf* $f_L(l)$ over all [possible starting waypoints in the area. The final result of these operations provides the transition length L of nodes moving according to the Rwp model on a disk [8] of radius a: in accordance of the following [1]:

iv. Transition Time

The transition time is the time taken by a node to move from one waypoint to the next waypoint. The above results are used on the transition length to calculate the stochastic properties of the transition time. The random variable and an outcome, are denoted by τ. It is considered as follows: $V_i = v = const\ \forall i\ and\ v > 0$. It implies that the speed of a node is constant during the entire movement. In this situation, it has

$$T = \frac{1}{v}L ---(12)$$

Hence, the expected transition time is

$$E\{T\} = \frac{1}{v}E\{L\} - - - (13)$$

and its *pdf* can be computed by

$$fr(\tau) = vf_L(v\tau) - - - (14)$$

With $E\{L\}$ and f_L taken from Equations (7) and (8) or Equations (10) and (11), respectively.

The speed of the node from a random distribution $f_v(\tau)$ at each waypoint (and then stays constant during one transition) instead of considering the speed as constant. hence T will be:

$$T = \frac{L}{V} - - - (15)$$

In this case, the random variable T is formed as a function $g(L, V) = L/V$ of two random variables L and V. In general, the expected value of a variable $g(L, V)$ can be expressed in terms of the joint *pdf* f_{LV}(l, v) as [15]

$$E\{g(L, V)\} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g\,(l, v)f_{LV}(l, v)dldv - - - (16)$$

In this case, L and V are independent, and thus their joint *pdf* is $f_{LV}\,(l, v) = f_L\,(l)f_V(v)$. The expected value can then be simplified as follows:

$$E\{T\} = E\{T\}\int_{v_{min}}^{v_{max}} \frac{1}{v} fv\,(v)dv - - - (17)$$

v. Transition Length and Duration

Transition Length and duration can be explained more in detail by using uniform speed distribution.

Uniform Speed Distribution

If uniform speed distribution is employed within $[v_{max}, v_{min}]$, the expected transition time is $fr\,(\tau) = \int_{v_{min}}^{v_{max}} vf_L\,(v\tau)fv\,(v)dv$ For $0 \leq \tau \leq \tau_{max}$ $with \frac{l_{max}}{v_{min}}$ and $\mathrm{fr}(\tau) = 0$ $- - -(18)$

$$E\{T\} = \frac{In\,(v_{max/v_{min})}}{v_{max} - v_{min}} E\{L\} - - - (19)$$

Note that $lim_{v_{max}\to v_{min}} \mathrm{E}\frac{\{\mathrm{L}\}}{v} E_L/v_{min}$ corresponds to Equation (17) for constant speed $\mathrm{v} = v_{min} = const.$ Further, it should also be noted that the expected time form $v_{min} = 0$ is undefined. This is very reasonable because if a node chooses V=0, the movement transition will take an infinite time. If the maximum speed can be expressed as a multiple of the minimum speed, that is, $v_{max} = kv_{min}$, with k > 1, it is obtained that

$$E\{T\} = \frac{In\,k\,E\,\{L\}}{k-l\,v_{min}} - - - (20)$$

vi. Time Duration

A node moves from one waypoint to another and then pauses from a certain time before changing direction in RWP model. The total T of an RWP period is then composed of a movement transition time T and pause time T_p. Now, it is possible to apply the above results for extending analysis for this case where a node rests a certain pause time in each way point as follows:

$$T = T + T_p - - - (21)$$

This linear combination of two independent random variables yields an expected value

$$E\{T'\} = E\{T\} + E\{T_p\} - - - (22)$$

and the *pdf*

$$f_{r'}(\tau') = \int_0^{\tau'} fr\,(\tau)fr_p(\tau' - \tau)d\tau\; for\; \tau' \geq 0 - - - (23)$$

The value of E{T′}represents the average time between two direction changes. Thus, the direction change frequency is given by $1/E\{T\}$ in unit 1/s.

vii. Spatial Node Distribution

This section investigates the spatial distribution of nodes in RWP model considering a rectangle or circular system area A. In earlier investigation, the distance and time between two consecutive waypoints was analyzed. However, these waypoints, (which represent the starting and ending points of a node's movement period), are uniformly distributed per definition. In practical it is studied only as the single node because all nodes move independently. The random variable considered can be represented as $X = f(X,Y)$ that denotes the Cartesian location of a mobile node in A at an arbitrary time instant t. A particular outcome of this variable is denoted by x. The spatial distribution of a node in terms of the probability density function is provided as stated below:

$$f_x(x) = f_{xy\,(x,y)}$$

$$= lim_{\delta\to 0}\ \Pr\left(\left(x-\frac{\delta}{2} < X \le x+\frac{\delta}{2}\right) \wedge \left(y-\frac{\delta}{2} < Y \le y+\frac{\delta}{2}\right)\right) / \delta^2 \quad ---(24)$$

Usually, a conversion to polar coordinates $R\sqrt{X^2+Y^2}$ and ∅ = arctan (Y/X) yields the joint distribution $f_{x(x)}$ over this subarea, that is,

$$\Pr(node\ in\ A') = P\ (X\ \in A')$$

$$= \int\int_{A'}^{1} f_{xy}\,(x,y)dA ---(25)$$

In Cartesian coordinates, the differential area element *dA* is given by $dA = dxdy$ and the resulting probability $\Pr\,(X \in A')$ can be interpreted as the percentage of time that a given mobile RWP node is located in the subarea A' during a long – run movement process with many transitions. Since the simulation is done with many mobile RWP nodes (n>1), it can also be considered as the ensemble average. As a consequence, $E\{n'\} = nPr$ (node in A') denotes the expected number of nodes located in A' at an arbitrarily chosen time instant.

viii. Stochastic Properties of Random Waypoint Model

Random Waypoint model as a discrete time stochastic process was described in Bettstetter et al.,2004. The transition length $L_i^{(j)}$ is defined as the distance covered by node j that moves from one waypoint to another during the *i*th epoch.

If the simulation field is a circular area with radius a. The probability density function of transition length L is

$$f_L(l) = \frac{8l}{2\pi a^2}\left[cos^{-1}\frac{1}{2a} - \frac{1}{2a}\sqrt{1-\left(\frac{1}{2a}\right)^2}\right] ---(26)$$

Correspondingly, the expected value of transition length L is

$$E[L] = \int_0^{2a} l f_L(l)dl = 0.905a\ ---(27)$$

and the variance of transition length L is

$$E[L^2] = \int_0^{2a} l^2 f_L(l)dl = \ a^2 ---(28)$$

Bettstetter et al.,2004 took a further step to derive the probability distribution of transition time as follow

$$f_T(t) = \int_{V_{min}}^{V_{max}} v f_L(vt) f_v(v) dv --- (29)$$

where $f_v(v)$ is the probability distribution function of movement velocity v and $f_L(l)$is the probability distribution function of transition length.

10. CONCLUSION

This work emphasized the documentation about NS2 simulator along with mobility model and performance metrics, work contains the performance comparison of the proposed protocols and existing protocol.

11. REFERENCES

[1] *Yan Sun; Yuhui Yao; Da Han; Phillips, C., "REAR: A radio environment adaptive routing protocol for CR Mobile Ad hoc Networks," in Telecommunications (ICT), 2015 22nd International Conference on , vol., no., pp.112-117, 27-29 April 2015*

[2] *Yaoran Zhang; Fei Song; Zhang Deng; Chao Li, "An energy-aware routing for cognitive radio ad hoc networks," in Information Science and Technology (ICIST), 2013 International Conference on , vol., no., pp.1397-1401, 23-25 March 2013*

[3] *Yi Song; Jiang Xie, " A QoS-Based Broadcast Protocol Under Blind Information for Multihop Cognitive Radio Ad Hoc Networks," in Vehicular Technology, IEEE Transactions on , vol.63, no.3, pp.1453-1466, March 2014*

[4] *Yi-Chi Chen; I-Wei Lai; Kwang-Cheng Chen; Wen-Tsuen Chen; Chia-Han Lee, "Transmission latency and reliability trade-off in path-time coded cognitive radio ad hoc networks," in Global Communications Conference (GLOBECOM), 2014 IEEE , vol., no., pp.1084-1089, 8-12 Dec. 2014*

[5] *Ying Dai; Jie Wu, "Boundary Helps: Efficient Routing Protocol Using Directional Antennas in Cognitive Radio Networks," in Mobile Ad-Hoc and Sensor Systems (MASS), 2013 IEEE 10th International Conference on , vol., no., pp.502-510, 14-16 Oct. 2013*

[6] *Yong Zhang; Junyi Wang; Jiming Lin, "A minimum delay link scheduling algorithm in vehicular ad-hoc networks," in Progress in Informatics and Computing (PIC), 2014 International Conference on , vol., no., pp.455-459, 16-18 May 2014*

[7] *Yuchen Guo; Yingchun Ma; Kai Niu; Jiaru Lin, "Transport capacity of cognitive radio ad hoc networks with primary outage constraint," in Wireless Communications and Networking Conference (WCNC), 2013 IEEE , vol., no., pp.3387-3391, 7-10 April 2013*

[8] *Z. Zhong, and T. Wei, Cognitive routing metric with improving capacity (CRM-IC) for heterogeneous ad hoc network. In Proceedings of international conference on information networking and automation (ICINA), Pages 271-274, 2010. Kunming, China.*

[9] *Zhiqing Wei; Zhiyong Feng; Qixun Zhang; Wei Li; Gulliver, T.A., "The asymptotic throughput and connectivity of cognitive radio networks with directional transmission," in Communications and Networks, Journal of , vol.16, no.2, pp.227-237, April 2014.*