Byungjin Cho

A Simulation study on Interference in CSMA/CA Ad-Hoc Networks using Point Process

Faculty of Electronics, Communications and Automation

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 31.12.2010

Thesis supervisor:

Prof. Riku Jäntti

Thesis instructor:

PhD. June Hwang

Aalto University
School of Science
and Technology

AALTO UNIVERSITY SCHOOL OF SCIENCE AND TECHNOLOGY	ABSTRACT OF THE MASTER'S THESIS

Author: Byungjin Cho

Title: A Simulation study on Interference in CSMA/CA Ad-Hoc Networks using Point Process

Date: 31.12.2010 Language: English Number of pages:11+79

Faculty of Electronics, Communications and Automation

Department of Communications and Networking

Professorship: Radio Communications Code: S-72

Supervisor: Prof. Riku Jäntti

Instructor: PhD. June Hwang

Performance of wireless ad-hoc networks is essentially degraded by co-channel interference. Since the interference at a receiver crucially depends on the distribution of the interfering transmitters, mathematical technique is needed to specifically model the network geometry where a number of nodes are randomly spread. This is why stochastic geometry approach is required.

In this thesis, we study about stochastic point processes such as Poisson Point Process, Matérn Point Process, and Simple Sequential Inhibition Point Process. The interference distributions resulting from the different point process are compared, and in CSMA/CA networks, point process's limitation issue such as the under-estimation of the node density is discussed. Moreover, we show that the estimated interference distribution obtained by Network Simulator 2, is different with respect to the different point process.

Even if there is the existence of gap between the distributions from the point processes and the simulator due to active factors, they all offer similar shape which follows a peak and an asymmetry with a more or less heavy tail. This observation has promoted an interest in characterizing the distribution of the aggregated interference with the Log-normal, Alpha-stable, and Weibull distributions as a family of heavy tail distributions. Even though hypothesis tests have mostly led to the reject of the null assumption that the interference distribution by the simulator is a random sample from these heavy tailed distributions except for the Alpha-stable distribution in high density. The hypothesis statistics systematically yield agreement on the choice of the better approximation. Moreover, the log probability process certainly makes it possible to reliably select the most similar heavy tailed distribution to the empirical data set on the variation of node density.

Keywords: Interference modelling, Stochastic geometry,
Poisson point process, Matérn point process, Simple Sequential Inhibition process.

Acknowledgements

This thesis is based on the work that was carried out in the Communications Laboratory, Aalto university, from June 2010 to December 2010.

First, I wish to express my sincere gratitude to my thesis supervisor Professor Riku Jäntti and instructor Ph.D June Hwang for giving me this opportunity to carry out this work. I would like to thank Professor Riku Jäntti for his continuous encouragement, professional advices, and constant support during the course of this thesis. I would like to thank my thesis advisor Ph.D June Hwang for his guidance throughout my work for this thesis. His wide knowledge and his logical way of thinking have been of great value for me. This work would not have been possible without his insightful comments and thoughts. His constant encouragement and his strive for excellence make working with him a rewarding experience. The discussion i had with Professor Riku Jäntti and Ph.D June Hwang, their comments and suggestions have greatly contributed to the quality of the thesis.

I wish to express my appreciation to my colleagues and friends, Cho's family, Sungin Cho, Kyunghyun Cho, and Eunah Cho for their help during the work. Their incredible support carried me through the roughest times and have helped to keep me sane with a diversity of liquor supply. I also would like to thank all my friends in Finland and in South Korea, for their genuine friendship.

Finally, the deepest gratitude goes to my family for their love and support. I haven't got enough words for thanking my parents for their encouragement during my whole life.

Otaniemi, 31.12.2010 Byungjin Cho

Contents

List of Abbreviations

NS	Network Simulator
PP	Point Process
PPP	Poisson Point Process
PPPmd	Poisson Point Process with modified density
MPP	Matern Point Process
SSI	Simple Sequential Inhibition
CSMA	Carrier Sensing Multiple Access
CA	Collision Avoidance
MAC	Medium Access Control
CCA	Clear Channel Assessment
RTS	Request-To-Send
CTS	Clear-To-Send
PDF	Probability Density Function
CDF	Cumulative Distribution Function
iff	if and only if
i.i.d	Independent and Identically Distributed
CS	Carrier Sense
DCF	Distributed Coordination Function
$chi2gof$	Chi-square goodness-of-fit
$kstest$	Kolmogorov-Smirnov Test
SINR	Signal to Interference and Noise Ratio
CLT	Central Limit Theorem
As	Alpha-stable distribution
Ln	Log-normal distribution
Wb	Weibull distribution
PLCP	Physical Layer Convergence Procedure

List of Symbols

$[k]$	The set $\{1, 2, ..., k\}$
$\mathbb{N}$	Positive integer
$\mathbb{R}$	Real number
d	Number of dimensions of the network
o	Origin in $\mathbb{R}^d$
B	a Borel subset of $\mathbb{R}$
$B(x, r)$	Ball of radius r centered at x
c_d	$\triangleq \pi^{d/2}/\Gamma(1 + d/2)$
$P(X)$	Probability of event X
$\mathbb{E}(X)$	Expectation of random variable X
$F_X(x)$	Distribution of random variable X (cdf)
$\Phi = \{x_i\} \subset \mathbb{R}^d$	Point Process
Φ_{PPP}	Poisson Point Process
Φ_{PPPmd}	PPP with modified density
Φ_{MPP}	Matern Point Process
Φ_{SSI}	SSI Point Process
Λ	Counting measure
λ	Node Density for Φ
λ_{PPP}	Node Density for Φ_{PPP}
$\lambda_{PPP-eff}$	Effective Node Density for Φ_{PPP}
λ_{PPPmd}	Node Density for Φ_{PPPmd}
λ_{MPP}	Node Density for Φ_{MPP}
λ_{SSI}	Node Density for Φ_{SSI}
h	Fading random variable
W	Thermal noise
χ^2	Chi-square
α	Characteristic exponent in Alpha-stable distribution
β	Symmetry parameter in Alpha-stable distribution
δ	Location parameter in Alpha-stable distribution
γ	Scale parameter in Alpha-stable distribution
m	Log mean parameter in Log-normal distribution
σ	Log scale parameter in Log-normal distribution

ν	Scale Parameter in Weibull distribution
k	Shape Parameter in Weibull distribution
r	Distance
η	Path loss exponent
D_{CS}	Carrier sense range
I_{CSTh}	Carrier sense threshold
e	Medium access indicator
P	Power emitted by a effective node
$l()$	Path loss function
μ	the mean fading is μ^{-1}
$\mathcal{L}$	Laplace functional of p.p Φ.
D_α	Critical value in Kolmogorov-Smirnov Test.
χ^2_α	Critical value in Chi-square.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Related Work

Spectral Resource reuse technique has been studied over the last decade to improve throughput capacity of wireless systems such as adhoc and sensor networks and cellular networks. While spectral resource reuse leads to capacity improvement, it causes co-channel interference, one of the key factors degrading overall performance in wireless communications. Therefore, a better understanding of the effects of interference on the performance of wireless communication system demanding a more aggressive utilization of spectral resources, for instance the newly emerged Cognitive Radio system, has become an important critical issue, motivating the development of more accurate interference models.

An interference model can be regarded as a mixture of different sub-models such as propagation model, interferers spatial distribution model, network operation model, and traffic model, each of which could be employed as a deterministic or random process, depending on the scenario considered. Since the accumulative interference generated by concurrent transmitting nodes overwhelmingly depends on nodes' location and the geometry information plays an important role in the interference modelling process, this thesis focuses on effective(active) interferers spatial distribution model, while the other components are taken into consideration under some assumptions which will be presented later.

Significantly unequal to infrastructure based wireless systems, there is high level of uncertainty in adhoc network which is formed by a number of nodes randomly spread over a large area. The position of the nodes are usually unknown, and hence the interference situation cannot be controlled by careful network planning. Therefore, a stochastic approach is required to properly describe the interference distribution.

In fact, a myriad of the spatial distribution models of the nodes have been suggested with using stochastic geometry tool which enable to make on average all realizations of the network whose nodes are placed according to some distribution, such as Poisson Point Process [1] which is analytically convenient and leads to some insightful results with its independence property.

However, in practice, this process is not accurate anymore for a CSMA/CA protocol in which a spatial correlation between nodes is introduced by medium access policy to avoid collisions. In recent works [2] [3], some researchers already made objection to this issue by modifying the initial Poisson process, so called hardcore models [4] where a minimum node separation is properly applied.

Nevertheless, they are still in their infancy due to some flaws [20] such as the spatial anomaly and thus an underestimation of the interference level. In [5], the use of an alternate model, referred to as the Simple Sequential Inhibition (SSI) point process was proposed. However, they do not provide the practical results by simulator.

1.2 Scope

In this thesis, we first introduce different point processes, and then compare the interference distributions resulting from these different ones. Moreover, we point out how different they are with practical simulation results induced by NS-2, and check whether the simulated interference could be extrapolated by considered distributions such as a log-normal or an alpha-stable distribution, thereby giving a benefit to engineers and researchers looking for appropriately approximated one to practical interference distribution as well as interference models in some particular scenario.

1.3 Contribution and Organization

This thesis is organized as follows.

Chapter 2 contains a description of the concepts that are essential for understanding the analysis that is to follow in the remaining of this thesis. The concepts of Point process and shot noise are introduced and a short introduction to heavy tailed distributions is presented.

In Chapter 3, the system model is introduced. Channel and MAC models used for this thesis are described. With results on Laplace functional of Poisson shot noise processes, some interference models for CSMA/CA network are introduced, and limitations for these models are discussed in detail.

Chapter 4 provides the results that are obtained by simulation tool, NS2. Based on the practical results, analysis for the interference models is discussed, and we test an extrapolation with the distribution resulting from NS-2.

Finally, conclusions and possible future research directions are discussed in Chapter 5.

Chapter 2

Preliminaries

In this chapter, we give a thorough background description on the concepts that are essential for understanding the analysis that follows in the remaining of this thesis. We begin in Section 2.1 with Point process and shot noise concept which are needed for stochastic geometry. Furthermore, in Section 2.2, we provide a short introduction to Alpha-stable, Log-normal, and Weibull distributions, which we will encounter in analysis of simulation result.

2.1 Spatial Point Process

In a case that every node has a single antenna and only a single channel is considered, interference is referred to as co-channel interference. In the statistical analysis of interference in wireless networks, the aggregate interference is the incoherent sum of individual interfering signals. The statistical characteristics of interference obviously depend on the statistics of the individual interfering signals. One of the main factors invoking the randomness of the individual interfering signals could be the distances between the location at which interference is received and interfering nodes. Since the distances influence on the mean power levels of interfering signals, these distances play an important role in the modelling process. Therefore, a stochastic model for the node locations (i.e., a spatial point process) is needed. Spatial point processes are the generalization of point processes indexed by time to higher dimensions, such as 2-D space. Stochastic geometry provides the tools to analyse important quantities such as interference distributions and link outages, and thus permits statistical statements about network performance [2, 3]. It shall allow us to focus on interference distributions in this thesis.

Formally, a point process Φ is viewed as a set of random points with a certain probability in a space E, where it is the Euclidean space $\mathbb{R}^d$ of dimension $d \geqq 1$. The intensity measure Λ of a point process Φ is equal to the average number of points in a set $B \subset \mathbb{R}^d$, and could be defined as:

$$\Lambda(B) = \mathbb{E}(\Phi(B)) \quad for \;\; Borel \; B. \tag{2.1}$$

Here are a few basic definition concerning point processes on some Euclidean space $\mathbb{R}^d$:

- Stationary: x_i and $x_i + x$ have same distribution.
- Isotropy: the same holds for all rotations about the origin.
- Motion-invariance: Stationary plus isotropy.

2.1.1 Poisson Point Process

Due to its analytical tractability and practical appeal in situations where transmitters and/or receivers are located or move around randomly over a large area, the Poisson point process (PPP) has been by far the most popular spatial model. Let Λ be a locally finite measure on some metric space E, the Euclidean space $\mathbb{R}^d$. A point processes Φ is Poisson on E if

- For all disjoint subsets $\{A_1, ..., A_n\} \subset \mathbb{R}^d$, the random variable $\Phi(A_1), ..., \Phi(A_n)$ are independent.
- For all sets $A \subset \mathbb{R}^d$, the random variables $\Phi(A)$ are Poisson.

The density of nodes in a unit area is λ, and so the average number of nodes in an area A is $\Lambda(A) = \lambda A$.

2.2 Shot Noise

Stochastic geometry combines the shot noise process with stochastic point processes which are drawn from a statistical distribution, most commonly the PPP as just discussed. We begin this section by reviewing some basic concepts on shot noise theory, considering the one dimensional case at first and the higher dimensional one for the interference modelling.

Let us consider a memoryless linear filter with stochastic impulse response function $f(h, t)$, where h is a random variable, and assume this filter is excited by a train of impulses at instants t_j derived from a one-dimensional Poisson Point process with intensity(density) λ.

As illustrated in Figure 2.1, the output $I(t)$ of the filter can be expressed as

$$I(t) = \sum_i f(h_j, t - t_j) \tag{2.2}$$

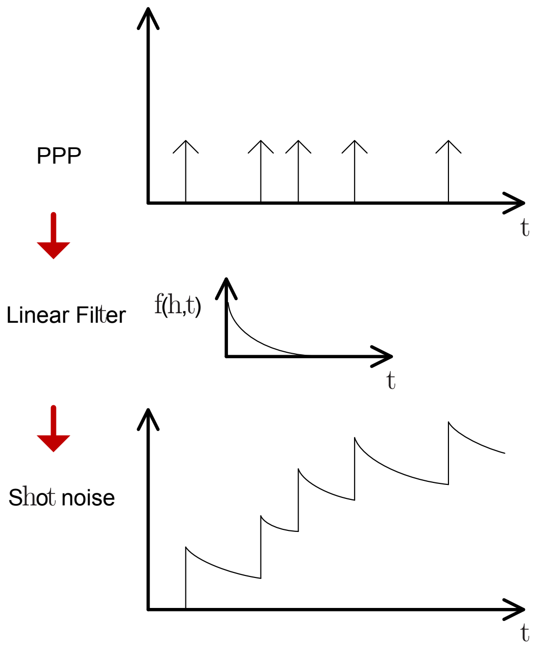


Figure 2.1: Shot Noise results from linearly filtering a PPP

where $\{h_j\}$ is independent of t_j and a random sequence, whose elements are drawn from a common distribution. The output $I(t)$ is said to be the shot noise associated with the Poisson Point process with intensity λ.

Through employing the concept above, we can build extension of this framework to higher dimensional networks. In order to model the location of transmitters in the wireless network, two-dimensional Poisson Point process defined in the Euclidean plane $\mathbb{R}^2$ is remarkably considered for the shot noise process related to the interference modelling.

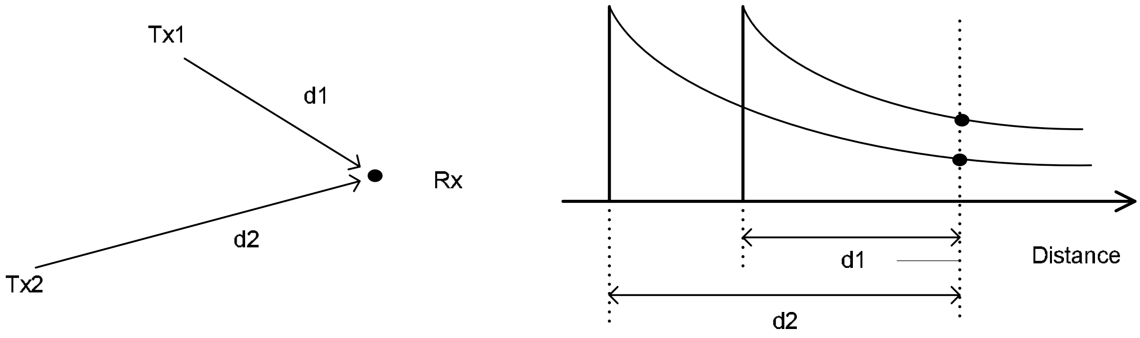


Figure 2.2: Interference modelling based on the shot-noise model, assuming deterministic path loss channel model

Next, the linear filter used in the 1-D case discussed above now models the path loss attenuation observed by interfering signals measured at receiver terminal located at the origin. Assuming a power-law decay model able to characterize the path loss attenuation, of the distance r between the transmitter and the receiver, with path loss exponent η, then the impulse response of the shot noise model is now written as $f(h,r) = hr^{-\eta}$. Therefore, the interference is modelled as

$$I = \sum_{i \in \Pi} h_i {r_i}^{-\eta} \tag{2.3}$$

where h is a random variable that models other effects of the propagation channel associated with the $i-$th interferer, and Π represents the Poisson point process defined by the interfering terminals. Figure 2.2 shows a physical interpretation of the interference modelling based on the shot noise model.

2.3 Heavy tailed distribution

It shall be stressed that any interference distribution observed for Poisson distributed interferers has the strongly skewed nature, unlike the classical observation [6]. Thus, some heavy tailed distributions are typical candidates for practical interference distributions to be considered in this thesis. In this section, we provide a short introduction to heavy tailed distributions including the Alpha-stable, Log-normal, and Weibull distributions we encounter in the chapter 4.

2.3.1 Alpha-stable distribution

The Alpha-stable distribution is a generalization of the Gaussian distribution. Alpha-stable distribution has heavier tails than the Gaussian distribution. Due to the lack of closed-form formulas for PDFs, the Alpha-stable distribution is often described by its characteristic function$\varphi(u)$, which is the Fourier transform of the PDF:

$$f_{Alphastable}(x;\alpha,\beta,\delta,\gamma) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \varphi(u)\exp(-ixu)du. \tag{2.4}$$

and the characteristic function is below:

$$\varphi(u) = \exp\{j\delta u - \gamma|u|^{\alpha}[1 + j\beta \text{sign}(u) w(u,\alpha)]\}$$

where

$$w(u,\alpha) = \begin{cases} \tan\left(\dfrac{\pi\alpha}{2}\right), & \text{for} \quad \alpha \neq 1 \\ \dfrac{2}{\pi}\log|u|, & \text{for} \quad \alpha = 1 \end{cases}$$

$$\mathrm{sign}(u) = \begin{cases} 1, & \text{for} \quad u > 0 \\ 0, & \text{for} \quad u = 0 \\ -1, & \text{for} \quad u < 0 \end{cases}$$

with the following definition for four parameters:

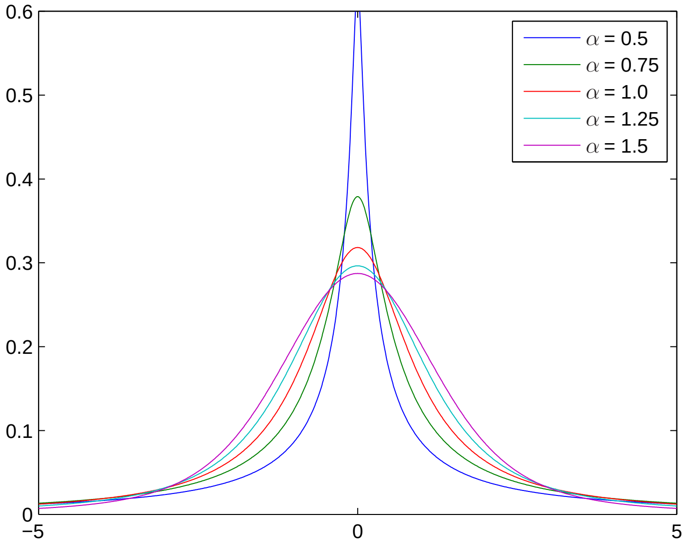


Figure 2.3: Skewed Alpha-stable densities, $\beta = 0$, $\delta = 0$, $\gamma = 1$

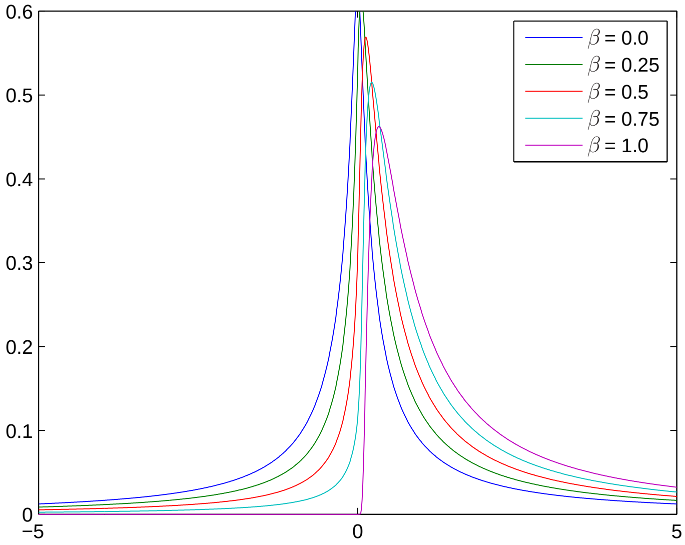


Figure 2.4: Skewed Alpha-stable densities, $\alpha = 0.5$, $\delta = 0$, $\gamma = 1$

- Characteristic exponent($\alpha \in [0, 2]$) measures the thickness of tails of the distribution; A small value of α will imply strong impulsiveness and considerable probability mass in the tails of the distribution, as shown in Figure 2.3.
- Symmetry parameter($\beta \in [-1, 1]$) sets the skewness of the distribution; when it is zero, the distribution is symmetric, as shown in Figure 2.4.
- Location parameter($\delta \in [-\infty, \infty]$)is similar to the mean in the normal distribution.
- Scale parameter($\gamma > 0$) is similar to the variance in the normal distribution.

The PDF of Alpha-stable random variable can be determined by taking the inverse Fourier transform of its characteristic function $\varphi(u)$ in (2.4). However, no closed-form expression exists for the Alpha-stable distribution, except for:

- For $\alpha = 2$, the Gaussian distribution.
- For $\alpha = 1, \beta = 0$, the Cauchy distribution.
- For $\alpha = 0.5, \beta = -1$, the Pearson distribution.

Stable distribution with $\alpha < 2$ differ from Gaussian ones in many ways. First, the tails decay like a power function. The smaller α, the slower the decay, and the heavier the tails. Moreover, Gaussian distributions are always symmetric around their mean, whereas the Alpha-stable distributions can exhibit arbitrary degrees of skewness. Due to these reasons, Gaussian distributions are not useful for modelling systems with high variability and random variables that are non-negative, while the Alpha-stable distributions are more flexible and do not exhibit such limitation.

The four parameters in the Alpha-stable distribution can not be directly derived from conventional statistical methods. Instead, several numerical methods, such as the maximum likelihood, have been developed to calculate these parameters. In this thesis, in order to estimate these parameters, we shall use Mark Veillette's Matlab stable distribution package [7].

2.3.2 Log-normal and Weibull distributions

Log-normal and Weibull are two most popular distributions for analysing skewed asymmetric data. The two distributions have several interesting properties and their PDFs can take different shapes.

The Log-normal distribution is based on the normal distribution. It describes a variable, x, where $\log(x)$ is normally distributed. It is valid for values of x which

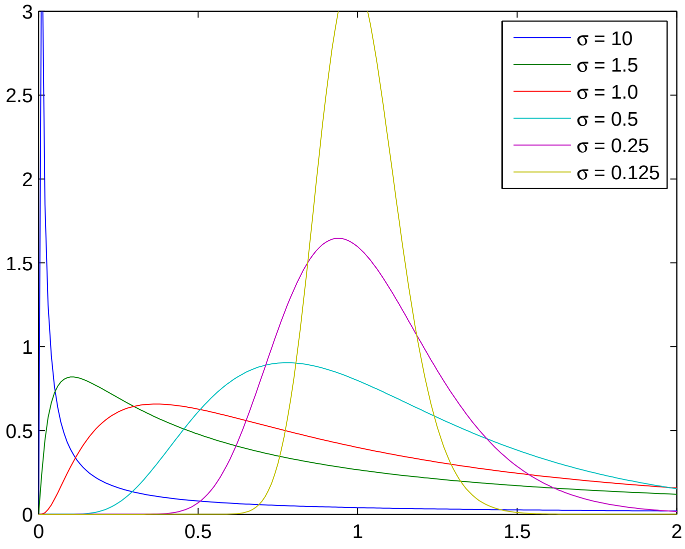


Figure 2.5: Log-normal densities, $m = 0$

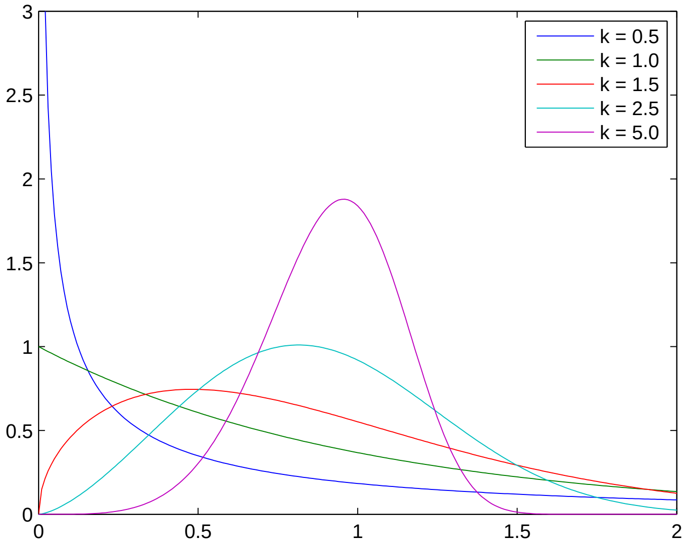


Figure 2.6: Weibull densities, $\nu = 1$

are greater than zero. The PDF of the Log-normal distribution is

$$f_{Lognormal}(x; m, \sigma) = \frac{1}{x\sigma\sqrt{2\pi}} \exp\left(-\frac{(\ln x - m)^2}{2\sigma^2}\right), x > 0. \tag{2.5}$$

where m is the location parameter or log mean, and σ is the scale parameter or log standard deviation. The location parameter is the mean of the data set after transformation by taking the logarithm, and the scale parameter is the standard deviation of the data set after transformation. The Log-normal distribution takes on several shapes depending on the value of the shape parameter. The Log-normal distribution is skewed right, and the skewness increases as the value of σ increases. Similarly, the density function of a Weibull distribution, with scale parameter ν>0 and shape parameter k>0 is

$$f_{Weibull}(x;\nu,k) = k\nu^{-k}x^{(k-1)}\exp\left(-\frac{x}{\nu}\right)^{k}, x > 0. \tag{2.6}$$

In chapter 4, $lognfit$ and $wblfit$ functions included in Matlab Statistical Toolbox are used to estimate the parameters for the Log-normal and Weibull distributions.

Chapter 3

SYSTEM MODEL

In this chapter, we present the system model, and apply some of the techniques introduced in the previous section to study the interference in ad hoc networks. This chapter is organized as follows: We begin with channel model and MAC model in Section 3.1 and 3.2 respectively. For interference model, Section 3.3 presents some results on Laplace functional of Poisson shot noise processes. Some Interference models for CSMA/CA network, that is, exclusively deployed nodes, are introduced and discussed in Section 3.4.

3.1 Propagation Channel Model

The two main propagation effects considered in this thesis are: (1) deterministic path loss and (2) small scale fading. These effects are described in the following.

Deterministic path loss models the attenuation suffered by a signal while traveling from the transmitter to the receiver. At a given location x, the power of a signal received from node y is given by $Pl(x; y)$. P is the transmission power of node y, and a function $l(x; y)$ gives the attenuation (path loss) from y to x in $\mathbb{R}^2$. For the deterministic path loss function, the following singular model will be our default assumption in this thesis. We will take for $\eta > 2$:

$$l(r) = l(x; y) = l(|x - y|) = \frac{1}{|x - y|^{\eta}} = r^{-\eta} \tag{3.1}$$

where η is the path loss exponent.

Propagation occurs through multiple paths between transmitter and receiver in a typical wireless communication environment, and several replicas of the transmitted signal reaching the receive antenna can be considered as a spatially non correlated and time variant process. These replicas combine with each other in a constructive or destructive way, resulting in a received signal with rapid envelope fluctuation. Several classical statistical distributions are commonly used to describe the envelope (or power) of the received signal. In this thesis, the amplitude fading is assumed to be the Rayleigh distribution. The functional form of the Rayleigh fading distribution is

$$Pr(a) = \frac{a}{\sigma^2}\exp\left(\frac{-a^2}{2\sigma^2}\right) \tag{3.2}$$

where a is the envelope amplitude of the received signal, and σ^2 is the pre-detection mean power of the multipath signal. When dealing with received signal powers, the power fading variable is used by denoting $P = a^2$, exploiting that the probability density function for the power is

$$Pr(P) = \frac{1}{\bar{P}}\exp\left(-\frac{P}{\bar{P}}\right) \tag{3.3}$$

where P stands for power and $\bar{P}$ is the mean of P, that is, the received signal power by path loss which is exponentially distributed with unit mean.

3.2 Medium Access Control

IEEE 802.11 DCF is considered for the CSMA/CA protocols [25]. The medium access control (MAC) protocol defines the rules for assessing the radio medium. The mechanism to determine whether or not the medium is busy is called CCA (Clear Channel Assessment). In this thesis, CCA is performed according to the following mode:

- *Energy above threshold.* CCA shall report a busy medium upon detection of accumulative signal power above a threshold called carrier sense threshold, I_{CSTh}. In this case, it could be expressed as $\sum_i P_i l(r_i) \leq I_{CSTh}$ where r_i is the distance from the effective node i to the sensing node, and the random parts of noise is not considered.

Given a carrier sense threshold, I_{CSth}, the corresponding carrier sense range, D_{CS}, is defined as the minimum distance allowed between two concurrent transmitters. In practical simulation by NS-2 which shall be introduced in next Chapter, CCA *Energy above threshold* Mode above is performed so that aggregate interference power value is evaluated with a given I_{CSTh}. In fact, D_{CS} is a non-deterministic parameter that can significantly affect the MAC performance in ad hoc networks. It balances the trade-off between the amount of spatial frequency reuse and the likelihood of packet collision. Therefore, it should be carefully chosen, based on network parameters such as network topology, traffic pattern, and transceiver power.

However, the deterministic carrier sense range is used e.g. in [9] [11] by exploiting the one-to-one mapping between a I_{CSTh}, and a D_{CS}. There is a further attempt to find the optimum carrier sensing range to maximize the throughput in [10]. For simplicity, point processes which shall be introduced in this Chapter employ the deterministic D_{CS} which can be expressed as:

$$D_{cs} = \left(\frac{P_0}{I_{CSTh}} \right)^{1/\eta} \tag{3.4}$$

where P_0 is transmission power and D_{CS} is carrier sense range.

3.3 Interference Models for PPP spatial node distribution

Nodes are deployed randomly at positions specified by a Poisson distribution. For the space $\mathbb{R}^2$, the probability that the area A in a PPP has n number of nodes is

$$Pr(N = n) = \frac{(\lambda A)^n}{n!} \exp(-\lambda A) \tag{3.5}$$

where N is the number of points and λ is the intensity(also called density) measure for the infinite planar. Then, we obtain the interference from nodes deployed by the PPP with node density λ_{PPP} which are the number of the potential and effective transmitters on the plane $\mathbb{R}^2$. Node geometry for PPP with $\lambda_{PPP-eff}$ follows:

$$\Phi_{PPP} = \{(X_i^{PPP}, (e_i^{PPP}, P_i^{PPP}))\}, \tag{3.6}$$

where we have the following.

- $\{i\}$ denotes a node's index.
- $\{X_i\}$ denotes the location of the node i.
- $\{e_i\}$ denotes the medium access indicator of the node i; e_i=0 means that the node is a potential receiver. e_i=1 means that the node is a effective transmitter. The random variable e_i are independent, with $Pr(e_i = 1) = p_{eff}$. The effective factor p_{eff} is the probability that a node has a packet to be transmitted at some time instant. Then we can express that $\lambda_{PPP-eff} = \lambda_{PPP} p_{eff}$.
- $\{P_i\}$ denotes the powers emitted by the station whose e_i is 1; The random variable $\{P_i\}$ are assumed to be independently and identically distributed with exponentially distributed powers with mean $1/\mu$.

Suppose that interference sensed at a node located at $y \in R^2$ follows process Φ_{PPP}. Through using the concept of shot noise process introduced in Chapter 2 and the attenuation function shown in Subsection 3.1, the shot noise process of Φ_{PPP} can be represented as

$$I_{\Phi_{PPP}}(y) = \sum_{X_i \in \Phi_{PPP}} P_i l(|y - X_i|). \tag{3.7}$$

Except for the path loss $\eta = 4$ case, the closed form of the shot noise distribution is not known for the other values of the path loss. In [8], the mean total received power is known as :

$$\mathbf{E}[I_{\Phi_{PPP}}] = \mathbf{E}[1/\mu]\lambda_{PPP-eff} \int_{\mathbb{R}^2} l(|y|)\mathrm{d}y = \frac{2\pi\lambda_{PPP-eff}}{\mu} \int_0^\infty rl(r)\mathrm{d}r \tag{3.8}$$

and the Laplace transform is also known as:

$$\begin{aligned} \mathcal{L}_{I_{\Phi_{PPP}}}(s) &= \exp\left\{-2\pi\lambda_{PPP-eff} \int_0^\infty \frac{r}{1 + \mu/(sl(r))}\mathrm{d}r\right\} \\ &= \exp\left\{-\lambda_{PPP-eff} \left(\frac{s}{\mu}\right)^{(2/\eta)} K(\eta)\right\}, \end{aligned} \tag{3.9}$$

where $K(\eta) = \dfrac{2\pi^2}{\eta \sin(2\pi/\eta)} = \dfrac{2\pi\Gamma(2/\eta)\Gamma(1-2/\eta)}{\eta}$ and $\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt$ is the Gamma function.

For $\eta = 4$ case, the PDF and CDF exists. From $\mathcal{L}_f(s) = s \cdot \mathcal{L}_F(s)$, CDF [28] [8] is:

$$\begin{aligned} F_{I_{\Phi_{PPP}}}(t) &= \Pr(I_{\Phi_{PPP}} \le t) \\ &= \mathcal{L}^{-1}\left\{\frac{\exp\left\{-\lambda_{PPP-eff}\left(\frac{s}{\mu}\right)^{(1/2)} K(4)\right\}}{s}\right\} \\ &= \mathrm{efrc}\left(\frac{\frac{\lambda_{PPP-eff}K(4)}{\sqrt{\mu}}}{2\sqrt{t}}\right) = \mathrm{efrc}\left(\frac{\lambda_{PPP-eff}\pi^2}{4\sqrt{\mu t}}\right) \end{aligned} \tag{3.10}$$

by using the fact that inverse Laplace transform of $\dfrac{\exp(-a\sqrt{s})}{s}$, $a \geq 0$, is erfc $\left(\dfrac{a}{2\sqrt{t}}\right)$, where $\mathrm{erfc}(x) = \dfrac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2}dt$ is a complementary error function. Accordingly, PDF is:

$$f_{I_{\Phi_{PPP}}}(t) = \frac{d}{dt}F_{I_{\Phi_{PPP}}}(t)$$
$$= \frac{\exp\left(\frac{\pi^4 {\lambda_{PPP-eff}}^2}{16\mu t}\right)\pi^{3/2}\mu\lambda_{PPP-eff}}{4(\mu t)^{3/2}} \tag{3.11}$$

3.4 Interference Models for Exclusively Deployed Nodes

According to CSMA/CA, only one can get a chance to access the channel among multiple nodes within overlapped carrier sensing region. Each node should be survived or discarded according to whether the distance between the nearest effective nodes is greater than the carrier sense range. In this thesis, only effective nodes are considered for interference models.

3.4.1 Poisson Point Process with modified density

In order to model the carrier sense range, the hard-core point process, in which the constituent points are forbidden to lie closer together than a certain minimum distance, one of thinning functioned process of initial point process is used in [4]. Thinning function means to discard certain existed points according to a given policy. since the determination of surviving points depends on the relative distance to other points already determined to survive, this thinning has a dependent characteristic.

In this model, the eventual node geometry which we consider here follows:

$$\Phi_{PPPmd} = \{(X_i^{PPPmd}, (e_i^{PPPmd}, P_i^{PPPmd}))\} \tag{3.12}$$

with modified node density measure given as follows:

$$\lambda_{PPPmd} = \frac{1 - \exp(-\lambda_{PPP-eff}\pi {D_{CS}}^2)}{\pi {D_{CS}}^2} \tag{3.13}$$

where $PPPmd$ denotes PPP with modified density. D_{CS} is the carrier sensing range. Elements of this process are ${e_i}^{PPPmd} = 1$ and ${P_i}^{PPPmd} = P_i$ as in (3.6) respectively. For CDF [28] [8] of the new shot noise interference in $\eta = 4$ case, we replace

$\lambda_{PPP-eff}$ of (3.11) with this new node density, λ_{PPPmd}:

$$F_{I_{\Phi_{PPPmd}}}(t) = \text{erfc}\left(\frac{\lambda_{PPPmd}\pi^2}{4\sqrt{\mu t}}\right) \tag{3.14}$$

Accordingly, PDF is

$$f_{I_{\Phi_{PPPmd}}}(t) = \frac{d}{dt}F_{I_{\Phi_{PPPmd}}}(t) = \frac{\exp\left(\frac{-\pi^4\lambda_{PPPmd}{}^2}{16\mu t}\right)\pi^{3/2}\mu\lambda_{PPPmd}}{4(\mu t)^{3/2}} \tag{3.15}$$

3.4.2 Matérn Point Process

An alternate temporal sequence version of the Matérn point process could be advocated [1] [2] to model interference in CSMA/CA network. A process where a subset of transmitters has been already chosen, is needed for the measure of the interference at a given point. So, a temporal scheduling between the points is requred.

A temporal Matérn point process is defined as an iterative procedure that sequentially tries to place n non-overlapping discs of radius D_{CS} in the plane. Denote this process Φ_{MPP} where MPP means Matérn Point Process. In this subsection, we start with this point process built as follows:

$$\Phi_{MPP} = \{(X_i{}^{MPP}, (e_i{}^{MPP}, P_i{}^{MPP}))\} \tag{3.16}$$

Elements of this process are $e_i{}^{MPP} = 1$ and $P_i{}^{MPP} = P_i$ as in (3.6) respectively. Then, Φ_{MPP} is developed as follows:

- Let N be a positive integer.
- Let X_i, where $i = 1, ..., N$, be a sequence of random variables independently and uniformly distributed in finite observation, $B(O, R)$, the ball of radius R centered at O.
- Let $\Phi_{MPP}(i)$ be the set of points selected after i steps.
- At the i^{th} step, the point X_i is distributed and selected in $\Phi_{MPP}(i)$ if and only if none of the $i-1$ previous points, even the inactive ones, lies in the finite plane B_{X_i}, the ball centered at X_i with radius D_{CS}.
- After all of the N points are processed with discrete Poisson distribution, the procedure finally ends.
- The point X_1 is distributed first and systematically selected in $\Phi_{MPP}(1)$.

- The Matérn point process $\Phi_{MPP}(N)$ is built for $i \in [2, N]$ according to:

$$X_{(i)} \in \Phi_{MPP}(i), \quad iif|X_i - X_j| > D_{CS} \; \forall j \in [1, i-1] \tag{3.17}$$

At each step, a new node attempts to access the channel medium. If its distance to all other points is greater than the carrier sensing range, it succeeds. Otherwise, the node is kept inactive. However, this inactive node is involved in the selection process of the following points. Note that the selection of a new point as active node depends on all previous points, even the inactive ones. This illustrates a spatial anomaly of the Matérn process related to the fact that inactive nodes give an impact on the selection process. In the end, the fact that unselected nodes play a role in the selection process limits the node coverage to a portion of the plane. The consequence is an underestimation of the effective transmitters density in the network and so far an underestimation of the interference.

3.4.3 Simple Sequential Inhibition

In order to complement the flaws of the alternate temporal sequence version of the Matérn point process presented in previous subsection, we discuss another point process, the Simple Sequential Inhibition(SSI) point process, which is first introduced by Palasti [12] and seems to be more appropriate model for CSMA/CA networks.

In this subsection, we start with the point process built as follows:

$$\Phi_{SSI} = \{(X_i{}^{SSI}, (e_i{}^{SSI}, P_i{}^{SSI}))\} \tag{3.18}$$

Elements of this process are $e_i{}^{SSI} = 1$ and $P_i{}^{SSI} = P_i$ as in (3.6) respectively. Then, Φ_{SSI} is developed as follows:

- Let N be a positive integer.
- Let X_i, where $i = 1, ..., N$, be a sequence of random variables independently and uniformly distributed in finite observation, $B(O, R)$.
- Let $\Phi_{SSI}(i)$ be the set of points selected after i steps.
- At the i^{th} step, the point X_i is distributed and selected in $\Phi_{SSI}(i)$ if and only if none of the $i-1$ previous points, only the active ones, lies in the finite plane B_{X_i}, the ball centered at X_i with radius D_{CS}.
- After all of the N points are processed with discrete Poisson distribution, the procedure finally ends.
- The point X_1 is distributed first and systematically selected in $\Phi_{SSI}(1)$.

- The SSI point process $\Phi_{SSI}(N)$ is built for $i \in [2, N]$ according to:

$$X_{(i)} \in \Phi_{SSI}(i), \quad iif|X_i - Z| > D_{CS} \; \forall Z \in \Phi_{SSI}(i-1) \tag{3.19}$$

At each step, a new node attempts to access the channel medium. If its distance to all other points is greater than the carrier sense range, it succeeds. Otherwise, the node is kept inactive and no longer considered in the selection process, unlike the previous Matérn process one.

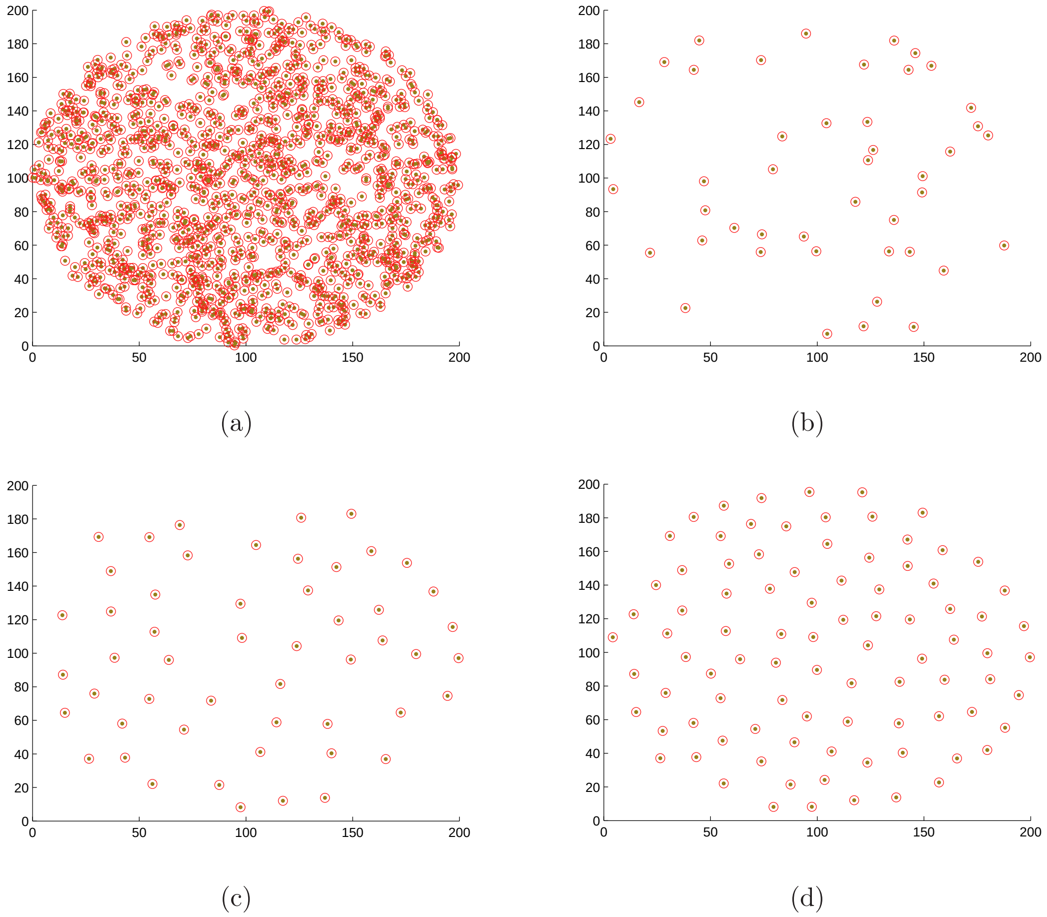
Figure 3.1 depicts samples of the point processes with a set of active nodes selected through different point process, PPP, PPPmd, MPP, and SSI, respectively with a large enough number of nodes. This obviously shows the under-estimated interference issue. In PPP model, nodes are selected regardless of the carrier sense range, leading to a very different interference distribution compared to the others with more suitable for CSMA/CA protocol. With the modified density in (3.14), PPPmd operates like PPP. The exclusion region with the given D_{CS} is applied to MPP and SSI models to discard the node attempting the channel medium. Since the selection process of SSI process just ignores inactive nodes dissimilar to the one of MPP, it is clearly different for the number of the selected node between MPP and SSI, hence the SSI model compensates for the main flaw of the Matérn model.

3.5 Challenge of Point Processes

In practice, CSMA/CA ad hoc networks [25] are formed by randomly deployed nodes including active transmitters giving theirs disturbance effects to their neighbours, and potential transmitters which could also be act as receivers.

From perspective of the point processes in introduced in previous sections, a point under consideration is simply viewed as an active transmitter already selected among all of the nodes. Therefore, a receiver or potential transmitter in practical situation is just a point that does not belong to the point process, Φ. In other words, the point processes do not reflect on the effective factor, p_{eff}, which is the probability that a node has a packet to be transmitted at some time instant.

So, there is undoubtedly difference between the consequences of the point process and practical scenario for IEEE 802.11 DCF based on the CSMA/CA protocol [25] which shall be implemented by NS-2 [24] [26] in the next Chapter. This challenge ultimately motivates us to show by simulation the difference between the interference distributions from the point processes and the simulated distribution by NS-2, and suggest that which distribution both interference consequences converge toward.



(a)

(b)

(c)

(d)

Figure 3.1: Sample of point process for $\lambda = 0.05$ nodes/m^2 $D_{CS} = 15$m. (a) PPP. (b) PPPmd (c) MPP. (d) SSI.

Chapter 4

Simulation Results and Analysis

The aim of this section is to validate how the practical results generated by NS-2 simulator match the different point processes introduced in previous Chapter. We then perform hypothesis test to check whether the empirical interference distribution could be extrapolated by some considered distribution. In Section 4.1, we first briefly show comparisons between the Matlab simulation results and the theoretical closed forms derived from PPP and PPP with modified density. On the basis of this certainty, we can extend our validation with NS-2 simulator. Simulation parameters and scenario used in NS-2 are introduced in Section 4.2. We carry out the comparison among the point processes in Section 4.3, and the existing gap between interferences from different processes and NS-2 is discussed in Section 4.4. Hypothesis test is executed in Section 4.5.

4.1 NS-2 Simulation Set-up

The simulations have been performed using NS-2 [24]. It is discrete event simulator and more common among researchers' community since it is an open- source simulator. The NS-2.34 version, the latest version [24] [26], is used for our experiments. Since cumulative SINR computation is offered by continuously tracking the sum of all reception power values of all frames arriving in parallel and of the noise floor, such significant information makes it possible to obtain the aggregate interference power level.

In simulator, $PowerMonitorthreshold$ is used to reduce the number of entries recorded in the interference list. For example, if a signal power from a transmitter is less than the value of this $PowerMonitorthreshold$ at a receiver, the receiver does not consider the transmitter as an interferer giving an effect on the aggregate interference power. In our simulation, we use a small enough value of the $PowerMonitorthreshold$ value so that all of nodes in a grid can be monitored and considered as interferers.

Preamblecapture feature is used in this simulation. While the receiver node is receiving the preamble and PLCP(Physical Layer Convergence Procedure) header of an earlier frame, if a new frame arrives at the receiver and it has sufficiently higher power which is *PreambleCaptureThreshold* value above the earlier one, the new frame can be picked. Likewise, while the receiver node is receiving the data of an earlier frame, if a new frame arrives at the receiver and it has sufficiently higher power which is *DataCaptureThreshold* value above the earlier one, then it immediately abandons the previous frame and attempts to decode the preamble and PLCP header of the new frame.

MAC header is transmitted with defined BPSK modulation(6Mbps), while the MAC data can be coded in a much higher modulation scheme, but in our simulation, it is also coded with BPSK modulation.

Figure 4.1 shows one of the scenarios in our simulation. In this thesis, the simulation descriptions are presented as follows:

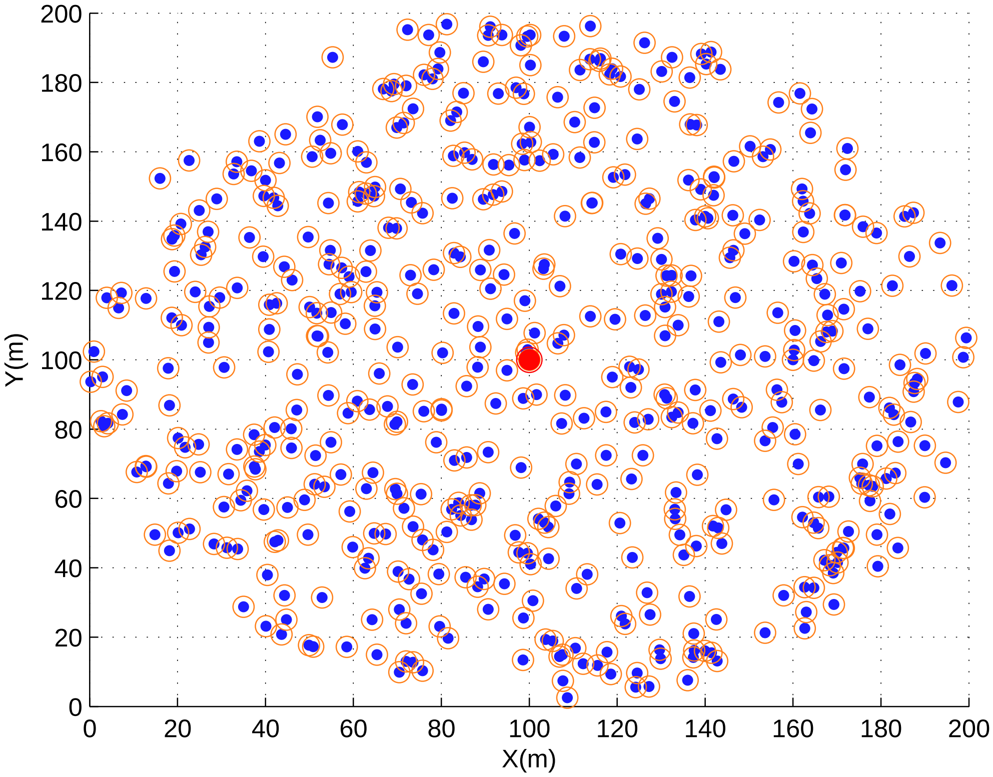


Figure 4.1: Topology Simulated NS2, measuring point at (100, 100)

Category	Parameter	Value
PHY 802.11a	Frequency	5.9GHz
	Transmission Power	1mW
	Channel bandwidth	20MHz
	Data rate(BPSK modulation)	6Mbps
	Power monitor threshold	0.12346pW
	Noise floor	1.9764nW
	Carrier sense threshold	6.25nW
	SINR preamble capture	4dB
	SINR data capture	10dB
MAC 802.11a	Slot time	9us
	SIFS time	16us
	ShortRetryLimit	7
	LongRetryLimit	4
	CWmin	15
	CWmax	1023
	Header Duration	20us
	Symbol Duration	4us
	RTS Threshold	0
Scenario	Grid	Circle using radius 100m
	Node density	0.0005, 0.005, 0.05 nodes/m^2
	Simulation iteration	100
	Number of measured data in each iteration	10,000
	Number of total measured data	1,000,000
	Simulation duration	15s
	Monitored time	5s
Application	Maximum packet number	10000
	Packet rate	6Mbps
	Traffic type	CBR(UDP)
	Data payload	1500Bytes/packet

Table 4.1: Simulation Configuration values

- For simplicity, we limit our simulation to single-hop transmission.
- For the CSMA/CA protocol, IEEE 802.11a DCF [25], CCA operation is performed according to the aggregated power level monitored in NS-2.34. If any detected power level is above the carrier sense threshold I_{CS}, CCA shall report a busy medium.
- For the wireless channel, Rayleigh fading effect is considered with the mean of power P in (3.3). In order to generate random value for this, we use the random number generator implemented by the RNG class [29], defined in $s/rng.h$ of NS2-34.

- By putting 0 into the value of RTS threshold, the RTS/CTS handshake is automatically operated regardless of packet size which is 1500 Bytes with CBR in this thesis. The sending channel rate of the source node and packet rate are 6 Mbps, thereby making the traffic saturated.

- Wireless nodes are uniformly distributed in area of circle with radius R 100m. "Uniformly distributed" means that all regions of the shape are equally likely to be selected by the random number generator. More formally, the probability of a number falling in a particular region is proportional only to the area of the region. For this, let a node on a circle with R centered at origin be defined by $(r\cos(\theta), r\sin(\theta))$ where r is the distance [0, R] from the origin and θ is the angle [0, 2π]. The values of r and θ are generated by uniform function in the RNG class [29]. This implementation is illustrated in Appendix B.

- Transmission procedures are operated amongst themselves according to randomly generated traffic scenario which is presented in Appendix B. All of the generated nodes are assigned to start transmission to each destination at a random time between 0 and 1 *second* until the maximum number 10000 of packets is transmitted or the simulation time 15 *second* is ended.

- The monitoring node (Red point in Figure 4.1) is additionally fixed at (100,100) to simply measure the aggregate power level of concurrent signals resulting from other nodes' transmissions. The measuring operations occur at "*Drop*" event [26]. In perspective of this monitoring node, all of the concurrently transmitting packets are seen as interference signals, since no node transmits a packet to the monitoring node as the transmitter's destination in our simulation.

- The whole simulation runs 100 times, but for each iteration, we extract the interference power level information within the specific time duration in order to ensure the case all of the nodes start to participate into the transmission.

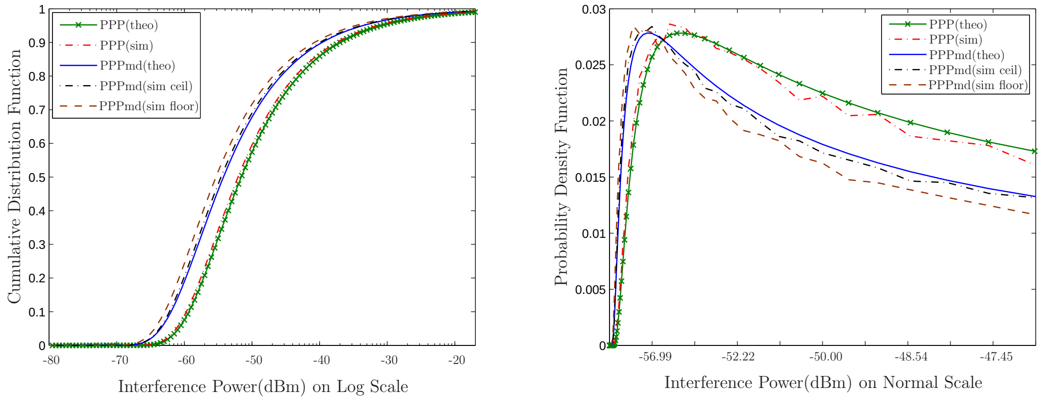
- For satisfying node distribution with a given node density, λ, at each iteration, the number of nodes in the area of a circle with radius R is generated as a random variable with poisson distribution (3.5) with parameter $\lambda\pi R^2$ at each iteration. The implemented code is shown in Appendix B.

- We sample 100,000 data from the measured total data, 1,000,000, from NS-2 simulation,. Half of this sampled data is used for plotting the PDF or CDF distributions of NS-2 result, and the other half is used for estimating the parameters of a considered distribution which shall encounter later as a fitting process.

Parameters used in our simulation are summarized in Table 4.1 with scenario descriptions. All the simulations presented later follow these parameters.

4.2 Theoretical and Experimental values of Interference distribution

As discussed in previous Chapter, the exact closed forms of the shot noise distribution for PPP and PPPmd can be only obtained, when the path loss η is 4. Using the inverse Laplace transform technique, we can derive the CDFs and PDFs from (3.11), (3.12), (3.15), and (3.16). In this Section, we compare the values from the theoretical closed form distribution with experimental results generated by Matlab, thereby providing the reliable foundation to compare with the practical ones based on NS-2 in the following Section.



(a) (b)

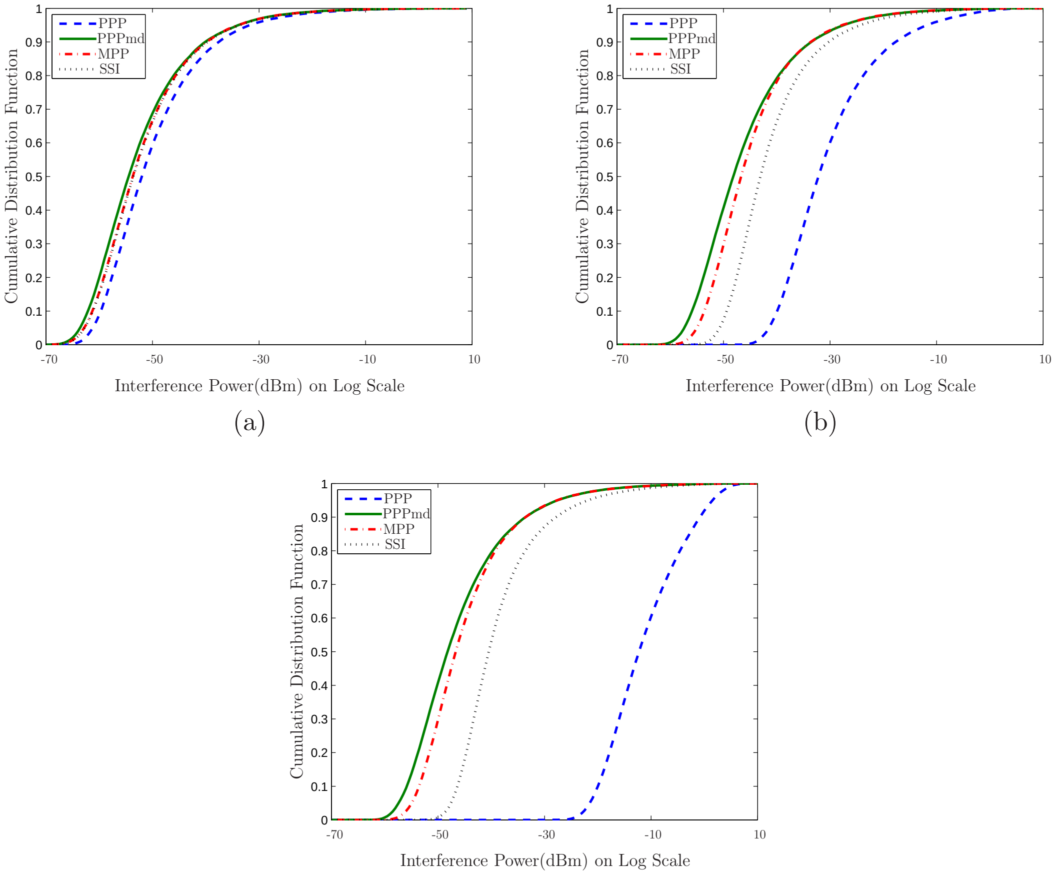
Figure 4.2: Theoretical results, and Experimental results in Matlab for $\lambda_{PPP} = 0.0005$ nodes/m^2 and $\lambda_{PPPmd} = 0.0375$ nodes/m^2 when I_{CSTh}= 6.25 nW. (a) CDF. (b) PDF.

In Figure 4.2, we have plotted the PDFs and CDFs of the interference. For simplicity, theo denotes the results from the closed forms, and sim denotes the simulation results from Matlab.

When it comes to PPPmd, we consider the modified node density measure given in (3.14) with the carrier sense range, D_{CS}. In order to obtain the real integer value from the modified node density, the number of nodes which is used in Matlab simulation, we use the ceiling($\lceil\rceil$) and floor($\lfloor\rfloor$) functions. We note that the theoretical result for PPPmd is very close to the experimental results using the respective number of nodes obtained by the ceiling and floor functions. Theoretical result and experimental one for PPP are also seen to be same. Since the obtained density by ceiling fucntion offers slightly more accurate result with theoretical one, we use the result from node density using ceiling function for PPPmd from the next Section.

4.3 Process Comparisons

In this Section, we compare PDFs and CDFs of interferences of the 4 point process models, PPP, PPPmd, MPP and SSI, and NS-2 simulation. The interference distributions for different node densities are shown as well.



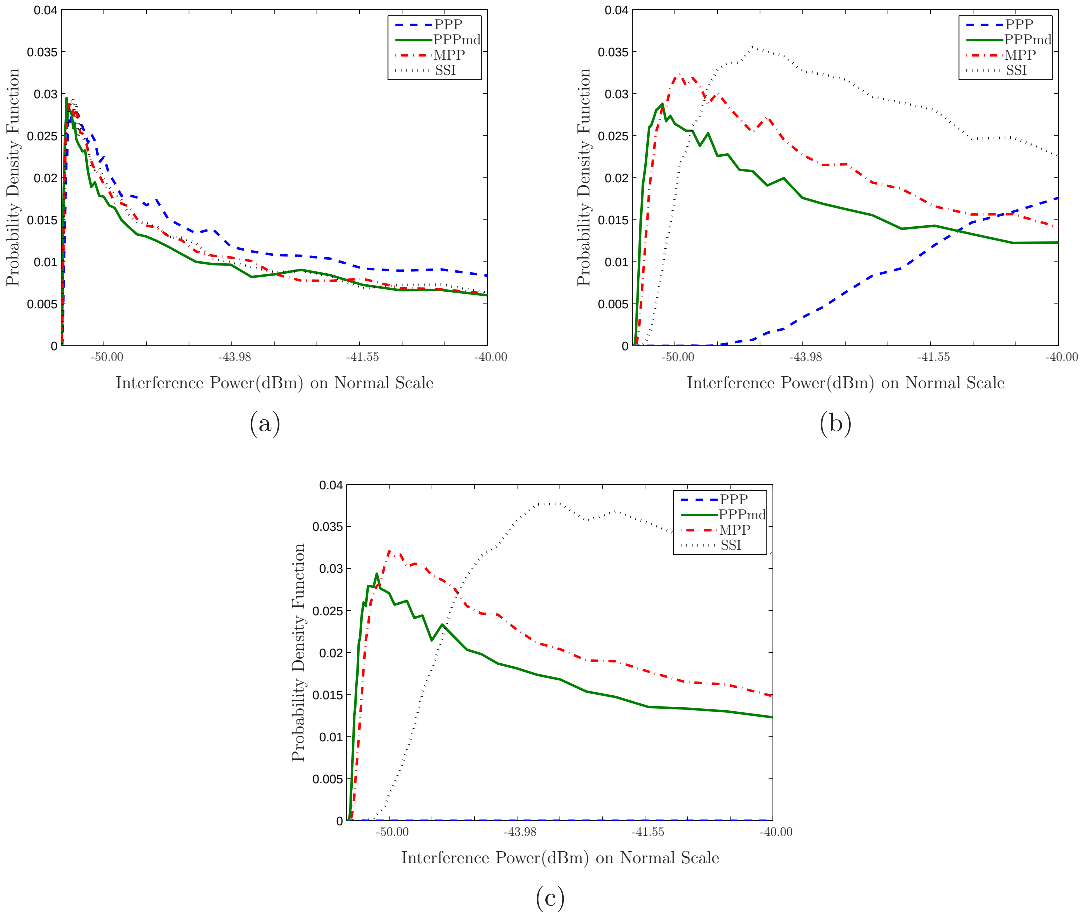
(a)

(b)

(c)

Figure 4.3: CDFs for the interference distributions of different processes, when I_{CSTh}= 6.25 nW. (a) $\lambda = 0.0005$ nodes/m^2. (b) $\lambda = 0.005$ nodes/m^2, (c) $\lambda = 0.05$ nodes/m^2.

In Figures 4.3 (a) to (c), we compare the interference CDFs along with density variation for the point processes. For low λ, all of the point processes offer similar interference distributions one another. More precisely and analytically, they all tend to follow the distribution of poisson point process. Since the possibility that some nodes reside within the radius, D_{CS} of one node is relatively very low, thereby making CSMA/CA scheme considering any dependency between the different transmission location appears to operate independently like Aloha scheme. As node density gets larger, interference distributions among different point processes start to differ more and more. For highest λ, the interference distribution by PPP



(a)

(b)

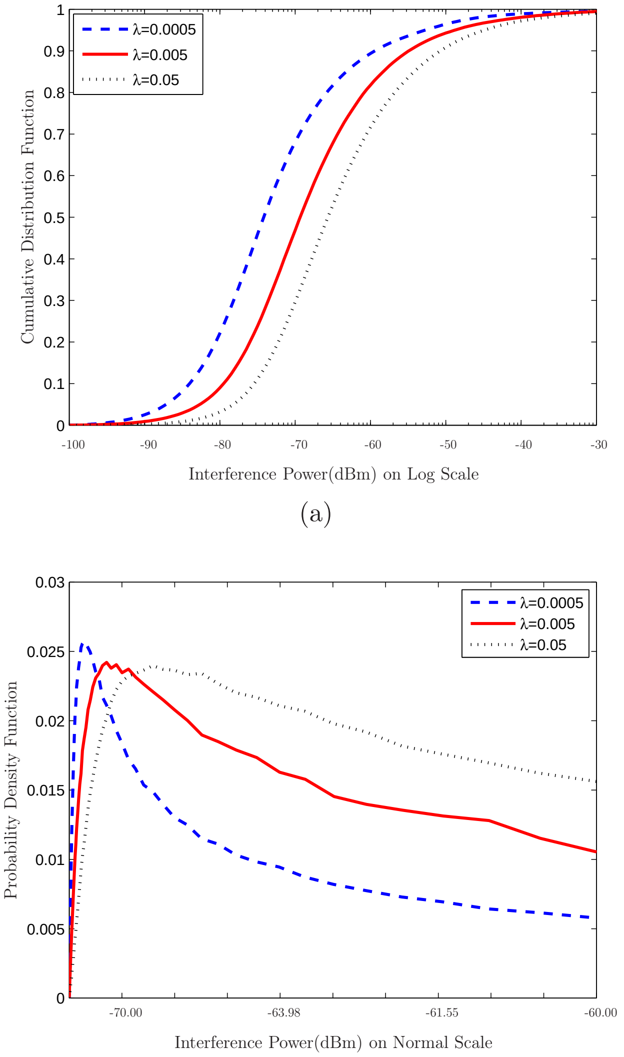
(c)

Figure 4.4: PDFs for the interference distributions of different processes, when I_{CSTh}= 6.25 nW. (a) $\lambda = 0.0005$ nodes/m^2. (b) $\lambda = 0.005$ nodes/m^2, (c) $\lambda = 0.05$ nodes/m^2.

becomes far from the other point processes, since it still consider all of the effective nodes unlike other processes.

In Figures 4.4 (a) to (c), the interference PDFs are plotted on a normal scale. Here we can see that PDF's peak point for each process is affected by a variation of density. For very sparse network, all of the processes offer similar shapes one another. In denser case, among the different point processes, there is huge dissimilarity regarding the variance of the distribution as well as peak point difference. Especially the difference of MPP and SSI interference distributions get large in more dense case, resulting from the phenomenon in which SSI makes an effort to resolve the flaw of MPP which underestimates the number of concurrent transmitters as discussed in previous chapter. Accordingly, the interference value of SSI's peak point is larger than of MPP's peak point.

Through Figure 4.5 summarily showing the interference distribution by NS-2, we



(a)

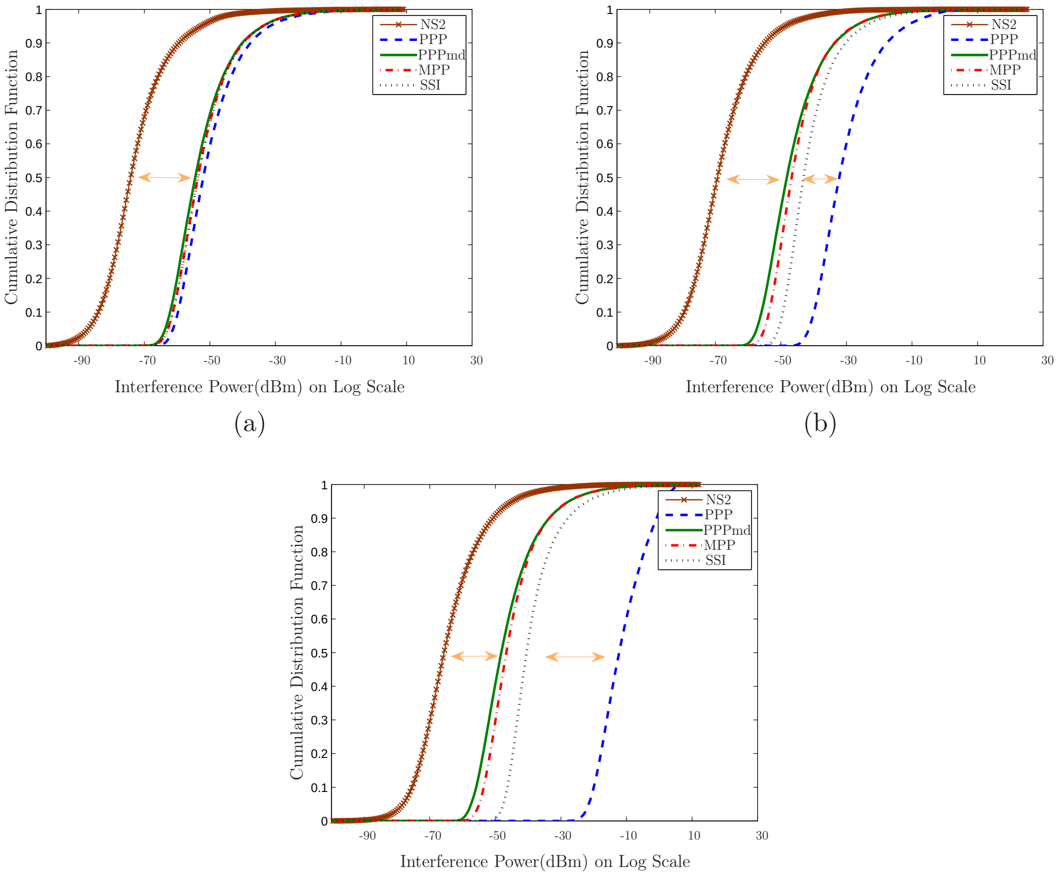
(b)

Figure 4.5: NS-2 simulation-based interference distribution for different values of λ, when I_{CSTh}= 6.25 nW. (a) CDF. (b) PDF.

can confirm that the practical interference distribution also varies with the different density.

4.4 Difference with NS-2 results

In Figure 4.6, the practical result by NS-2 and interference distributions by the point processes are illustrated. Note that the interference distributions generated by the various point processes are significantly different with the practical results by NS-2. The main reason for the existence of gap is that node densities used in the inter-



(a)

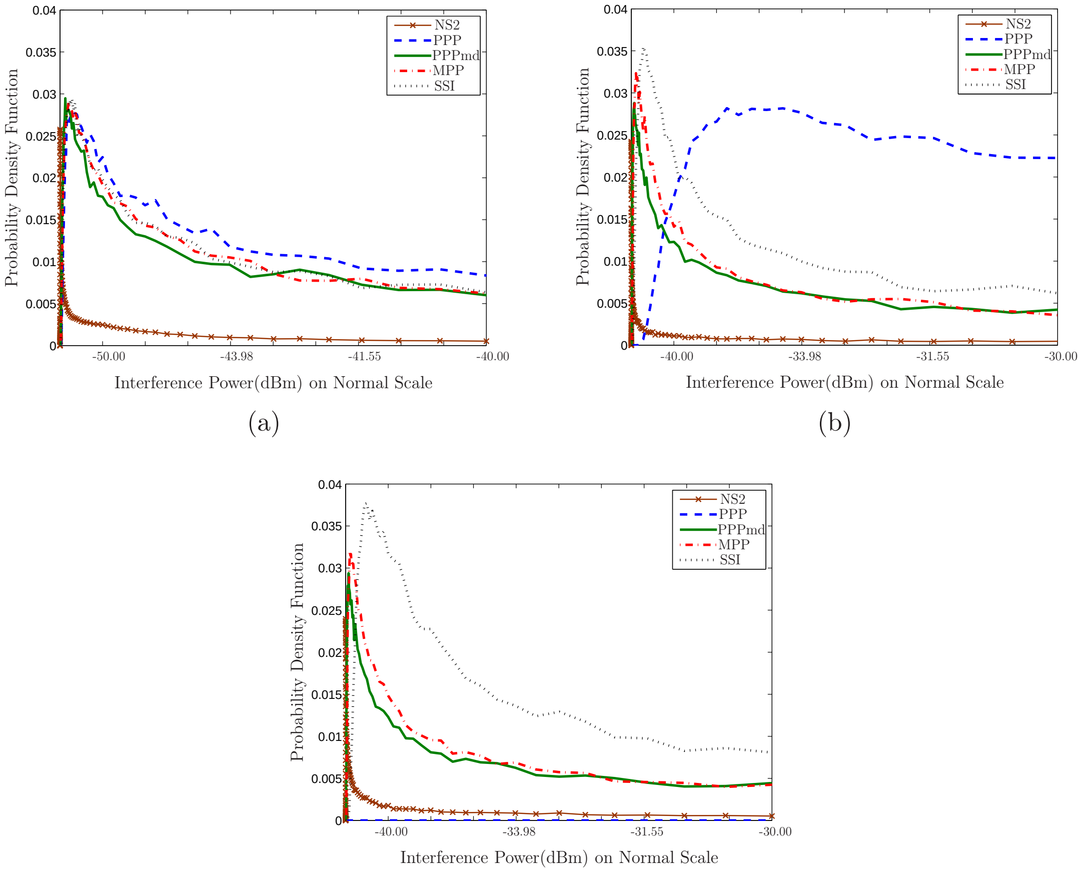
(b)

(c)

Figure 4.6: Gap between NS-2 simulation-based interference distribution and Point processes-based interference distributions, when I_{CSTh}= 6.25 nW. (a) $\lambda = 0.0005$ nodes/m^2. (b) $\lambda = 0.005$ nodes/m^2, (c) $\lambda = 0.05$ nodes/m^2.

ference models are assumed to be effective node density representing only effective transmitters, not potential ones. Thus, the point process models simply concern the aggregated interference power level which decays with distance from interferes, whereas the NS-2 simulation result is derived by nodes' effective characteristic factor, p_{eff} which practically results from some parameters related to MAC layer and Application layer, such as initial contention window size, contention window size, retry-limit, traffic generation rate, and so on. Furthermore, different from the fact that the deterministic D_{CS} is employed by (3.4) in the presented point process models, D_{CS} is nondeterministic in NS-2 simulation. Thus, D_{CS} could be decided by the CCA mode(*Energy above threshold*) mentioned in Section 3.2 with a carrier sense threshold I_{CSTh}.

Nevertheless, since they all offer the similar shape: a peak and an asymmetry with a more or less heavy tail depending on the point process in Figure 4.7, it is enough worth of notice. This observation absolutely contradicts a classical assumption in



(a)

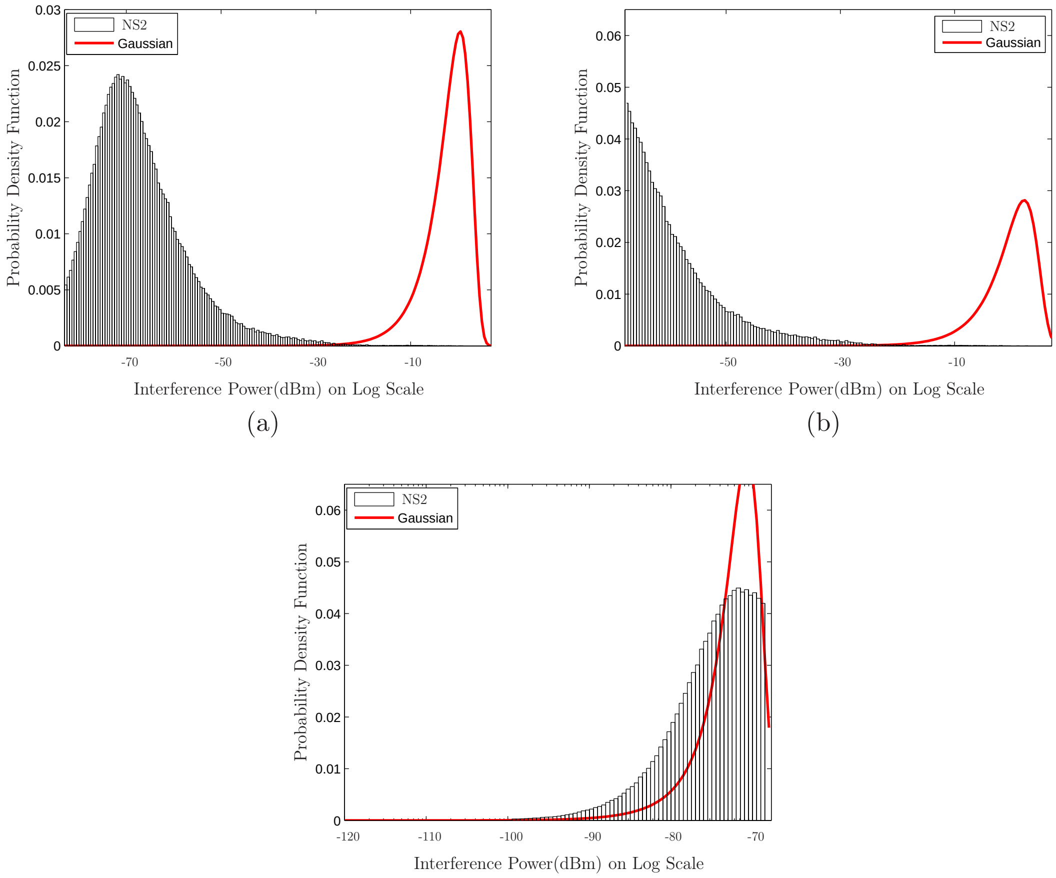
(b)

(c)

Figure 4.7: PDF shapes with asymmetric and heavy tailed behaviour, when I_{CSTh}= 6.25 nW. (a) $\lambda = 0.0005$ nodes/m^2. (b) $\lambda = 0.005$ nodes/m^2. (c) $\lambda = 0.05$ nodes/m^2.

the signal processing community where the interference is generally considered to be Gaussian [6]. Some researcher in [30] indicates that when considering the aggregate interference, as a number of interferers in a network goes to infinity, the Gaussian distribution is a proper approximation for the distribution of the aggregate interference. However, we should pay attention to the fact that in the case where some of the interferers are dominant, the central limit theorem(CLT) [23] is not valid anymore [31], even if the number of interferers may be large.

Figure 4.8 shows the PDFs of the interference distribution by NS-2, compared with a Gaussian distribution on Log Scale. Here, using $normfit$ function in MATLAB toolbox, we fit the Gaussian distribution to the NS-2 result. According to Figure 4.8 (a), there is obvious difference between the NS-2 result and Gaussian distribution. Figure 4.8 (b) also shows that the distribution forming tail part of the overall distribution is also hugely different from a Gaussian distribution, due to the heavy tailed characteristic of the NS-2 result, which is mainly produced by the dominant interferers around a node. In fact, different from other literatures [32] [33], in our



(a)

(b)

(c)

Figure 4.8: Comparison between NS-2 simulation-based histograms for the interference and Gaussian fit, when I_{CSTh}= 6.25 nW and $\lambda = 0.005$ nodes/m^2. (a) Overall. (b) Dominant. (c) No dominant

scenario, a measuring node centered at origin without any sense range or exclusion range, just measures the concurrently transmitting signal power without discarding any node. Therefore, many interferers around the node are dominantly able to give an effect on aggregate interference with strong power. Ultimately, our scenario makes it possible to offer an opportunity generally aware of interference distribution at any arbitrary point from a geometry, not a conditionally restricted situation in which a measuring node also participate in communication.

In addition, it is observed from Figure 4.8 (c) that the interference distribution excluding the dominant values seems to approximate the similar shape to the Gaussian distribution. It means that even if not an infinite random field of interferers, limited geometry decided by finite planer or PowerMonitor threshold value in PowerMonitor module of the NS-2 simulator, is considered, the distribution without the dominant interference is close to a Gaussian one.

4.5 Statistical Significance Test

In this section, the difference between point process models and NS-2 result is objectively proven by value of statistics which simply offer the trend in variation of node density. We test several hypotheses that the interference by NS-2 conforms log-normal, Weibull or Alpha-stable distribution, and further obtain log-probabilities to find the most similar distribution to the empirical one.

4.5.1 Hypothesis Checking Technique

We use the Matlab numeric computing environment and its Statistics Toolbox, a collection of tools supporting general statistical functions to curve fitting. The techniques of hypothesis checking consist of two basic procedures. First, values of distribution parameters are to be estimated by analysing experimental sample. Second, the null hypothesis that experimental data have a particular distribution with certain parameters should be checked. To perform hypothesis checking itself, the $kstest2$ and $chi2gof$ functions are used in the subsection 4.5.2 and 4.5.3, respectively or both.

4.5.2 process comparison with NS2 result

In this subsection, the $kstest2$ function, $h = kstest2(\boldsymbol{x_1}, \boldsymbol{x_2})$, performs a two-sample Kolmogorov-Smirnov test to compare the distributions of the values in the two data vectors, $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ which are interference results by NS-2 and one of the point processes. The null hypothesis is that two vectors are from the same continuous distribution. The alternative hypothesis is that they are from different continuous distributions. Result h is equal to "1" if the hypothesis can be rejected, or "0" if we cannot reject that hypothesis. The function also returns the p-value which is the probability that the null hypothesis can not be contradicted. The test statistic value is used to decide whether or not the null hypothesis should be rejected. In our work, we reject the hypothesis if the test is significant at the 5% level(p-value less than 0.05).

Let the first sample be the NS2 results with CDF $F(\boldsymbol{x_1})$ and squentially put one of the different point processes into the second sample group with CDF $G_i(\boldsymbol{x_2})$, in order to compare NS2 result with one of the processes.

$$H_0 : F = G_i \quad \text{vs.} \quad H_1 : F \neq G_i, \qquad \forall\ i \quad \in \quad PointProcesses.$$

The statistic, shown in Table 4.2, is used to compare the fitness of NS-2 with each process. For large sample size, the approximate critical value D_α is 0.0086 with $\alpha = 0.05$ and n= 50000, according to this equation:

λ (nodes/m^2)	0.0005	0.005	0.05
Critical Value (D_α)	8.601395e-03	8.601395e-03	8.601395e-03
G_1 PPPmd	7.981769e-01	8.225285e-01	7.368599e-01
G_2 MPP	8.105696e-01	8.605087e-01	7.908243e-01
G_3 SSI	8.124147e-01	9.110835e-01	9.021974e-01
G_4 PPP	8.391473e-01	9.640765e-01	9.931161e-01

Table 4.2: Kolmogorov-Smirnov test statistic for different density, when I_{CSTh}= 6.25 nW

$$D_\alpha = c(\alpha)\sqrt{\frac{n_1+n_2}{n_1 n_2}}$$

where n_1 and n_2 are the sample sizes of $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$, respectively, and the coefficient $c(\alpha)$ is given by the table below.

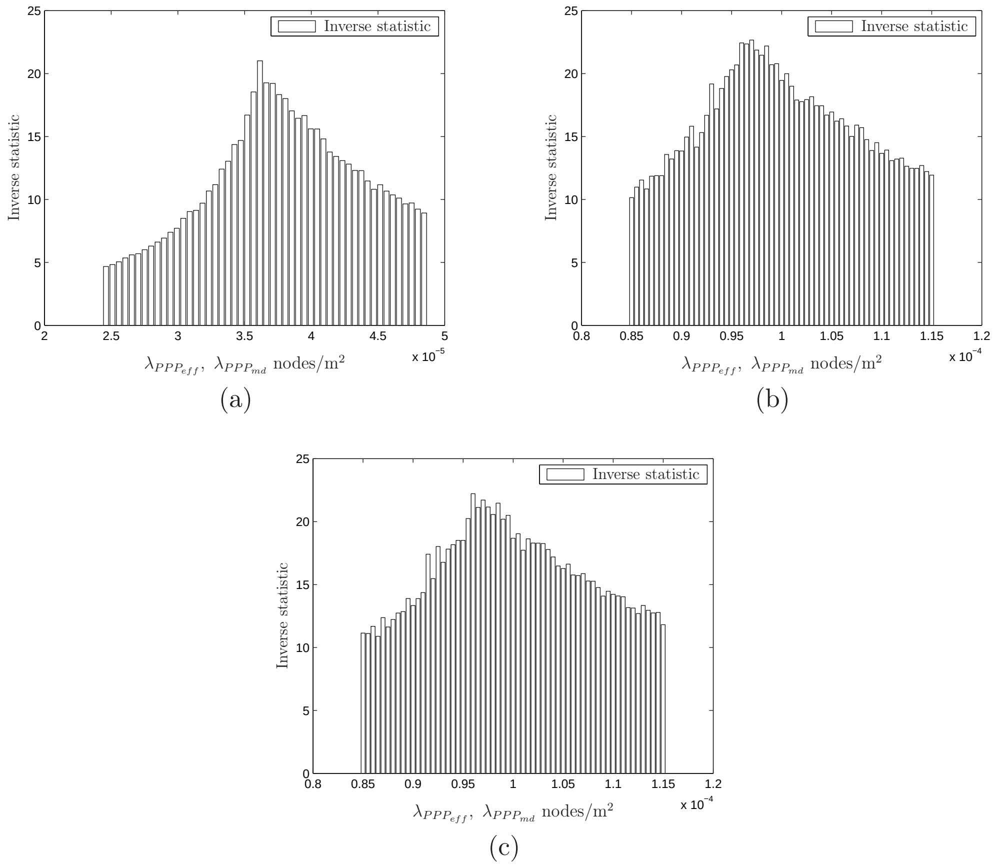
α	0.10	0.05	0.025	0.01	0.005	0.001
$c(\alpha)$	1.22	1.36	1.48	1.63	1.73	1.95

Table 4.3: the coefficient $c(\alpha)$ values for α in [22]

Since all of the Kolmogorov-Smirnov test statistic values are larger than the critical value D_α, we reject all of the null hypotheses that interference distributions from NS-2 and each of Point processes are from the same continuous distribution.

As mentioned in the previous section, since each of precesses and NS-2 result belong to the different mean and variance, more precisely, unlike NS-2 results, a family of point process models considered in this thesis do not reflect on the realistic factors, such as some parameters in MAC layer for CSMA/CA system, it is no wonder that the rejection resulting from these statistic values is induced. Therefore, the absolute value above cannot suggest which point process gives the best fit to realistic interference distribution. Nevertheless, in the density variation's effect point of view, we can observe that the difference between statistics is larger, as the node density is larger. It could lead to the interpretation that interference distributions of point process models are absolutely affected by the density variation.

However, there is no knowing whether or not the rejections for all of the null hypotheses are only caused by the p_{eff} not taken into consideration. In the context, it is required to find an effective node density offering the best fit or approximation to the interference distribution by NS-2, thereby making it possible to reliably compare NS-2 result to process results. Figure 4.9 shows the inverse statistic values obtained in Kolmogorov-Smirnov test making comparison between NS-2 simulation-based interference distribution with λ and interference distributions of PPP and



(a)

(b)

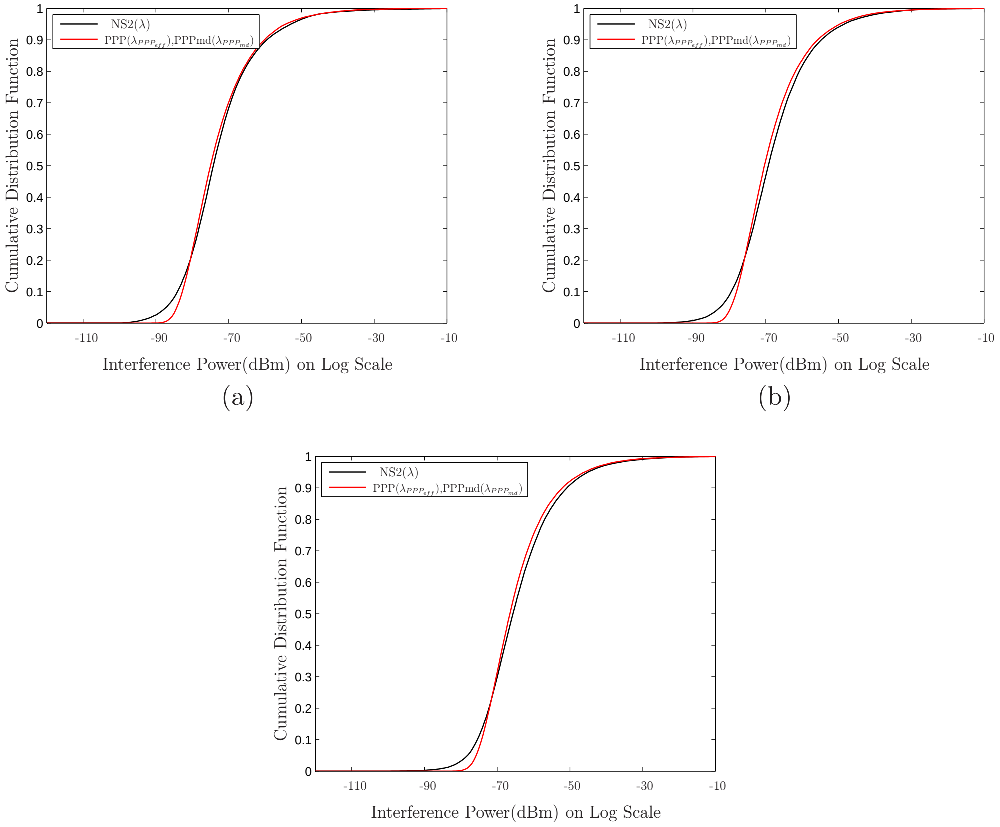
(c)

Figure 4.9: Kolmogorov-Smirnov test Inverse statistics of PPP with $\lambda_{PPP_{eff}}$ and PPPmd with $\lambda_{PPP_{md}}$ for NS-2 simulation-based interference distribution and, when I_{CSTh}= 6.25 nW. (a) $\lambda = 0.0005$ nodes/m^2. (b) $\lambda = 0.005$ nodes/m^2. (c) $\lambda = 0.05$ nodes/m^2.

NS-2 (λ)	PPP ($\lambda_{PPP_{eff}}$)	PPPmd ($\lambda_{PPP_{md}}$)	PPPmd ($\lambda'_{PPP_{eff}}$)	Inverse -statistics	Statistics	D_α
0.0005	3.65e-05	3.65e-05	5.39e-05	20.0227	4.94e-02	8.601395e-03
0.005	6.25e-05	6.25e-05	9.39e-05	19.4401	5.14e-02	8.601395e-03
0.05	9.60e-05	9.60e-05	1.48e-04	22.2124	4.50e-02	8.601395e-03

Table 4.4: $\lambda_{PPP_{eff}}$ and $\lambda_{PPP_{md}}$ offering the best approximation to NS-2 simulation-based interference distributions, and corresponding statistic values, when I_{CSTh}= 6.25 nW

PPPmd with $\lambda_{PPP_{eff}}$ and $\lambda_{PPP_{md}}$, respectively, which are varied with interval density $5e-07$. At every interval density, in order to sample interference values, the inverse CDFs for (3.10) and (3.14) are computed with uniform random numbers (50000) between 0 and 1.



(a)

(b)

(c)

Figure 4.10: NS-2 based interference distribution with λ, and PPP and PPPmd interference distributions with the optimal $\lambda_{PPP_{eff}}$ and $\lambda_{PPP_{md}}$, respectively, when I_{CSTh}= 6.25 nW. (a) $\lambda = 0.0005$ and $\lambda_{PPP_{eff}}, \lambda_{PPP_{md}} = 3.65e-05$. (b) $\lambda = 0.005$ and $\lambda_{PPP_{eff}}, \lambda_{PPP_{md}} = 6.25e-05$. (c) $\lambda = 0.05$ and $\lambda_{PPP_{eff}}, \lambda_{PPP_{md}} = 9.60e-05$.

In Table 4.4, the effective node densities, $\lambda_{PPP_{eff}}$ of PPP and $\lambda_{PPP_{md}}$ of PPPmd, offering the best approximation to NS-2 results with different λ are summarized. $\lambda'_{PPP_{eff}}$ denotes $\lambda_{PPP_{eff}}$ term of PPPmd in (3.13) to avoid confusion with $\lambda_{PPP_{eff}}$ of PPP. Figure 4.10 shows NS-2 based interference distribution with λ, and PPP and PPPmd interference distributions with the optimal $\lambda_{PPP_{eff}}$ and $\lambda_{PPP_{md}}$, respectively. According to the Kolmogorov-Smirnov test statistic values larger than D_α, we objectively reject all of the null hypotheses that interference distributions from NS-2 with different λ and interference distributions of PPP and PPPmd with corresponding $\lambda_{PPP_{eff}}$ and $\lambda_{PPP_{md}}$, respectively, are from the same continuous distribution. It means that PPP and PPPmd models do not offer the perfect fit to the practical interference distribution induced by NS-2. Yet, it could be clearly observed that the effective node densities, $\lambda_{PPP_{eff}}$ and $\lambda_{PPP_{md}}$, do not get large much, while it increases 10 times in the λ. This observation reflects on the fact [34] [35] that even though the saturation traffic in Table 4.1 could increase a case where a node is ready to transmit a data to a receiver, the CSMA/CA scheme introduces a reasonable transmission probability with RTS/CTS mode and exponential random backoff scheme used in NS-2 simulation, making an effect on effective node geometry.

4.5.3 Extrapolation Analysis

As discussed in the previous section, the interference distribution presents a peak and a heavy-tail. Several distributions such as Alpha-stable, Log-normal, Weibull, K-distribution, Gamma or Laplacian distribution have been proposed to model this heavy-tailed distribution. We here focus on the Alpha-stable, Log-Normal, and Weibull distributions.

In this subsection, in order to execute more reliable measurement, with Kolmogorov-Smirnov test, the $chi2gof$ function, additionally performs a Chi-square goodness-of-fit test of default null hypothesis that interference result by NS-2 is a random sample from the Alpha-stable, Log-Normal, or Weibull distribution with estimated parameters in Table 4.6 for each distribution. The test is performed by grouping the data into bins, calculating the observed and expected counts for those bins, and computing the chi-square test statistic

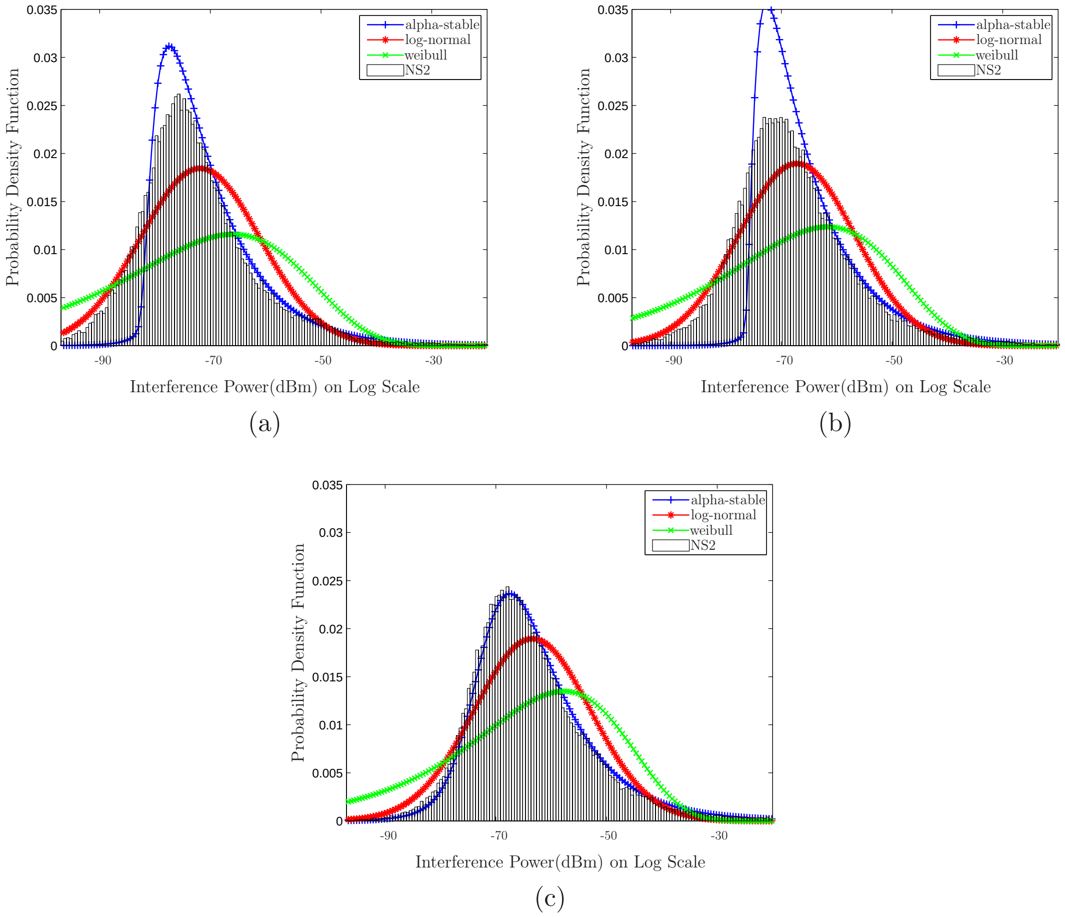
$$\chi^2 = \sum_{i=0}^{N} \frac{(O_i - E_i)^2}{E_i},$$

where N is the number of samples, O_i are the observed counts in each bin, and E_i are the expected counts in each bin. In order to get comparable quantities, the number of bins, $nbins$ is 10 as default, and the N is 50,000 for NS-2 result.

In Figures 4.11(a) to 4.11(c), we plot the interference PDFs on log scale for NS-2 result to compare with the Alpha-stable, Log-normal and Weibull distributions. As we can see, the Weibull distribution does not follow the shape of PDF by NS-2 for all of the different density cases, while the Log-normal and Alpha-stable distributions offer a rough or close approximation of the practical interference distribution by NS-2 for the different density of nodes.

Chi-square	Alpha-stable	Log-normal	Weibull
Degree Freedom	5	7	7
Critical Value ($\chi_\alpha{}^2$)	11.071	14.067	14.067
$\lambda = 0.0005$	5.148101e+03	4.108520e+03	2.859666e+04
$\lambda = 0.005$	2.242956e+04	2.734669e+03	2.270185e+04
$\lambda = 0.05$	1.107098e+01	3.107760e+03	1.945508e+04

Table 4.5: Chi-square test statistics for different heavy tailed distributions with NS-2 result, when I_{CSTh}= 6.25 nW



(a)

(b)

(c)

Figure 4.11: Heavy tailed distributions and NS-2 simulation-based interference histograms for different density, when I_{CSTh}= 6.25 nW. (a) $\lambda = 0.0005$ nodes/m^2. (b) $\lambda = 0.005$ nodes/m^2. (c) $\lambda = 0.05$ nodes/m^2.

Kolmogorov-Smirnov	Alpha-stable	Log-normal	Weibull
Critical Value (D_α)	8.601395e-03	8.601395e-03	8.601395e-03
$\lambda = 0.0005$	1.182000e-01	8.932000e-02	1.860800e-01
$\lambda = 0.005$	1.681400e-01	6.822000e-02	1.564000e-01
$\lambda = 0.05$	8.240000e-03	7.222000e-02	1.442800e-01

Table 4.6: Kolmogorov-Smirnov test statistics for different heavy tailed distributions with NS-2 result, when I_{CSTh}= 6.25 nW

Nevertheless, the χ^2 and Kolmogorov-Smirnov tests mostly lead to the reject of the null assumption that interference induced by NS-2 is a random sample from the considered heavy tailed distributions above. The χ^2 and the Kolmogorov-Smirnov statistics presented in Table 4.4 and 4.5 allow us to objectively compare the accuracy of the Alpha-stable, Log-Normal, and Weibull approximations.

We can see that the χ^2 and Kolmogorov-Smirnov statistics are larger than the critical value corresponding to each distribution and hypothesis test except for only the Alpha-stable distribution in dense case. Thus, we clearly conclude that all of the distributions qualitatively do not fit to the empirical result by NS-2 except for only the Alpha-stable distribution in dense case, even if the Log-normal distribution offers close approximation in the Figure 4.11. It should be, however, noted that the statistical values in both hypothesis tests for Log-normal distribution seem to be better than those for the Alpha-stable and Weibull distributions, when λ are 0.0005 and 0.005 nodes/m^2. According to the statistics of the hypothesis tests, the Alpha-stable distribution fits in extream case of node densities, and the Log-normal distribution is, for the other densities, the best approximation among these considered heavy tailed distributions.

In order to find a distribution most similar to the interference distribution induced by NS-2 among the Alpha-stable, Log-Normal, and Weibull distribution, we here concentrate on the probability of NS-2 data set with 50000 samples same as hypothesis tests. Loosely speaking, the probability for the data set is the joint probability of a particular set of data given the chosen probability model and set of parameters which has been already estimated from the fitting process in Matlab, as shown in Table 4.6.

Let $x_1, ..., x_n$ be an i.i.d. random sample from NS-2 data set, and for a given set of parameters, a probability density function Pr is given by

$$\mathrm{Pr}_{As}(x_1, x_2, ..., x_N|\boldsymbol{\theta}_{As}) = \prod_{i=1}^{N} \left[\int_{x_i-\epsilon}^{x_i+\epsilon} f_{As}(x_i|\boldsymbol{\theta}_{As})dx \right]$$

$$\mathrm{Pr}_{Ln}(x_1, x_2, ..., x_N|\boldsymbol{\theta}_{Ln}) = \prod_{i=1}^{N} \left[\int_{x_i-\epsilon}^{x_i+\epsilon} f_{Ln}(x_i|\boldsymbol{\theta}_{Ln})dx \right]$$

$$\mathrm{Pr}_{Wb}(x_1, x_2, ..., x_N|\boldsymbol{\theta}_{Wb}) = \prod_{i=1}^{N} \left[\int_{x_i-\epsilon}^{x_i+\epsilon} f_{Wb}(x_i|\boldsymbol{\theta}_{Wb})dx \right]$$

where N is the number of sample, f is the continuous pdf, and $0 < \epsilon << 1$. Here, we call As Alpha-stable, Ln Log-normal and Wb Weibull distributions. Due to numerical considerations, the logarithm of the probability density functions is employed because the use of the summation avoids numerical overflows.

$$\log[\mathrm{Pr}_{As}] = \sum_{i=1}^{N} \log \left[\int_{x_i-\epsilon}^{x_i+\epsilon} f_{As}(x_i|\boldsymbol{\theta}_{As})dx \right]$$

Distributions	Parameter $\boldsymbol{\theta}$	$\lambda = 0.0005$	$\lambda = 0.005$	$\lambda = 0.05$
Alpha-stable	α	4.961760e-01	4.779910e-01	4.845367e-01
	β	9.158094e-01	9.144688e-01	1.000000e-00
	γ	2.152848e-11	6.521046e-11	1.466723e-10
	δ	1.213325e-12	1.542156e-11	-1.711938e-11
Log-normal	m	-2.354072e+01	-2.243617e+01	-2.156463e+01
	σ	2.449433e+00	2.469530e+00	2.398825e+00
Weibull	ν	2.352524e-10	6.641183e-10	1.585883e-09
	k	2.801167e-01	2.913005e-01	3.198153e-01

Table 4.7: Estimated Parameter Values used to Generate Random variable as a Function of the Distribution of NS-2 result, when I_{CSTh}= 6.25 nW

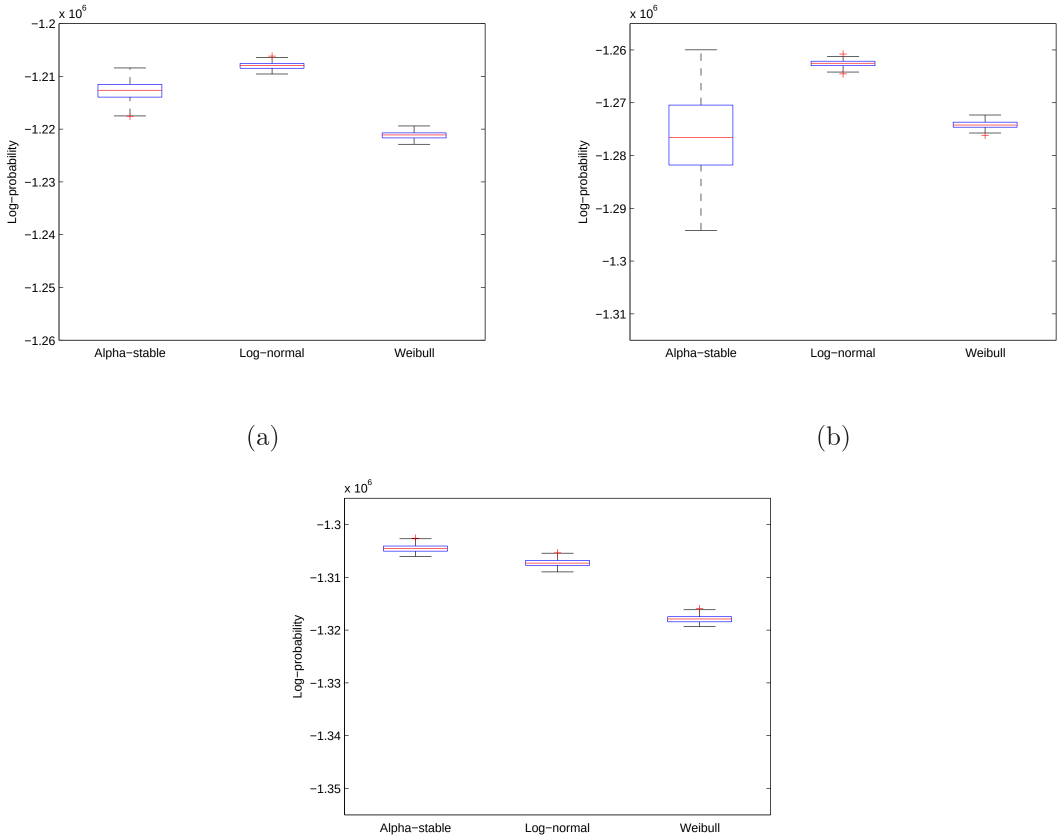
$$\log[\Pr_{Ln}] = \sum_{i=1}^{N} \log\left[\int_{x_i-\epsilon}^{x_i+\epsilon} f_{Ln}(x_i|\boldsymbol{\theta}_{Ln})dx\right]$$

$$\log[\Pr_{Wb}] = \sum_{i=1}^{N} \log\left[\int_{x_i-\epsilon}^{x_i+\epsilon} f_{Wb}(x_i|\boldsymbol{\theta}_{Wb})dx\right]$$

where N is the number of sample and $0 < \epsilon << 1$.

Figure 4.12 shows the log probability density of the NS-2 data set by 100 independent trials, further, confirms that the Alpha-stable distribution achieves the highest log probability density value of the NS-2, when λ is 0.05 nodes/m^2, and the log-normal distribution reaches the highest log probability density, when λ are 0.0005 and 0.005 nodes/m^2.

Table 4.7 shows the median value of the log probability density of the NS-2 sample data set over 100 repeated runs, along with the variation of node density. The Alpha stable and Log-normal distributions are comparable each other to NS-2 data set, whereas the Weibull distribution is relatively far from the NS-2 data set compared to the others. This observation makes it possible to select a distribution most similar to the NS-2 data set on the variation of node density.



(a)

(b)

(c)

Figure 4.12: For probability density values, the red line inside the box denotes the median, the edge of the box are 25-th and 75-th percentiles, and the whiskers are extended to the extreme value not considering possible outliers, when I_{CSTh}= 6.25 nW. (a) $\lambda = 0.0005$ nodes/m^2. (b) $\lambda = 0.005$ nodes/m^2. (c) $\lambda = 0.05$ nodes/m^2.

Distributions	$\lambda = 0.0005$	$\lambda = 0.005$	$\lambda = 0.05$
Alpha-stable	-1.212633e+06	-1.276548e+06	-1.304535e+06
Log-normal	-1.207953e+06	-1.262510e+06	-1.307296e+06
Weibull	-1.221095e+06	-1.274214e+06	-1.317877e+06

Table 4.8: Median value of Logarithmic probability density of the NS-2 sample data set over 100 repeated runs, given the set of parameters for Alpha-stable, Log-Normal, and Weibull distribution, when I_{CSTh}= 6.25 nW

Chapter 5

Conclusions and Future Work

In this chapter we are going to conclude the main results achieved in this thesis. Most importantly, some possible future research directions based on this thesis are discussed as well.

5.1 Conclusions

As interference results from the summation of signals issued by concurrent transmitters, it directly depends on the nodes location. Thus, the point process used to model concurrent nodes location is fundamental in any multi-hop wireless network research. Assuming the active transmitters are distributed according to a hardcore process, the PPP with modified density, MPP, and SSI are discussed for CSMA/CA networks where a busy medium detection depends on the aggregated power from all concurrent signals.

For very sparse networks, the PPP with modified density, MPP, and SSI, tend to be similar to the result of PPP, since the PPP does not consider any dependency between the transmitter location. For denser case, various point processes lead to different interference distributions. Noticeably, the difference between MPP and SSI gets large, due to the under-estimated density issue. In the practical interference distribution itself induced by NS-2, node density variation lead to shift of peak point.

We point out, however, that the interference distributions generated by the various point processes are significantly different with the practical results by NS-2. The existence of large gap is mainly due to the assumption that all of the point process models only consider multiple concurrently active node, whereas in NS-2, the active node, the probability of medium access of a typical node is not given a priori and it has to be determined by the CSMA/CA DCF policy. In addition, Kolmogorov-Smirnov test executed to ensure objective comparison with statistic values, result in the rejection of the assumption that interference distributions from NS-2 and each of point processes are from the same continuous distribution. Even under consideration of the effective node densities offering the best approximation to the interference distribution by NS-2, the hypothesis test leads to the rejection again. However, the statistics of the hypothesis test allow us to confirm that the CSMA/CA scheme introduces a reasonable transmission probability with RTS/CTS mode and exponential random backoff scheme, making an effect on effective node geometry.

Nevertheless, we have observed that all of the interference distributions by NS-2 and Processes all offer the similar shape which follows a peak and an asymmetry with a more or less heavy tail depending on each mean and variance. This observation absolutely contradicts a classical assumption in which the interference is generally considered to be Gaussian. We show that in the case where some of the interferers are dominant, the central limit theorem(CLT) is not valid any more, even if the number of interferers may be large. We further confirm that in a case which gets rid of dominant interference, the distribution has a tendency similar to the Gaussian shape.

Besides, this observation has promoted an interest in characterizing the distribution of the aggregated interference with the Log-normal, Alpha-stable, and Weibull distributions as a family of heavy tail distributions. Through the fitting plot, the empirical interference distributions by NS-2 are approximated by the Alpha stable and Log-normal distribution for the different densities of nodes. However, the χ^2 and Kolmogorov-Smirnov tests have mostly led to the reject of the null assumption that the interference result by NS-2 is a random sample from the Alpha-stable, Log-Normal, or Weibull distribution with estimated parameters except for the Alpha-stable distribution in high node density. The hypothesis statistics from these tests systematically yield agreement on the choice of the better approximation. Additionally, the log probability process with several iterations certainly makes it possible to more reliably select a distribution among the considered 3 kind of the heavy tailed distributions most similar to the NS-2 data set on the variation of node density. This approximation provides the opportunity to analytically study and characterize values such as SINR which is used to specify and ultimately compute the connectivity/coverage and the capacity/throughput.

5.2 Possible Future Work

Interesting research topics about spatial model based on stochastic geometry for interference are below

- to build a more realistic modelling to consider the concurrently transmitting node density without pre-assumption which most of point processes should employ. In order to make it possible, we are able to begin with transmission probability Biancchi introduced for MAC performance measurement.

- to derive new distribution closest to practical interference or find some trend of parameters representing for theoretical distribution along with various factors such as density, CS range, RTS/CTS usage, or channel parameters.
- to apply this spatial model for interference to different type of wireless networks such as Cognitive radio and Femtocells as well as Ad hoc networks. When it comes to Cognitive Radio, a primary consideration of a cognitive radio is the likelihood of interfering with a primary. The probability of this occurring must be held small, and it clearly depends on the spatial density and the typically unknown locations of the primary receivers. Stochastic geometry provides an essential toolkit for understanding femtocell deployments. A natural model for two-tier networks consisting of tier 1 base stations and tier 2 femtocells is to model the femtocell locations as a point process of density lf overlaying a regular grid of base stations.

Bibliography

[1] F. Baccelli, B. Blaszczyszyn, and P. Muhlethaler, "An Aloha protocol for multihop mobile wireless networks," *IEEE Trans. Information Theory,* vol.52, no.2, pp. 421- 436, Feburary 2006

[2] Y.-C. Cheng and T.G. Robertazzi, "A new spatial point process for multihop radio network modeling," *Proc. 1990 IEEE International Conf. Comm.,* pp.1241-1245

[3] A. Busson and G. Chelius, "Point processes for interference modeling in CSMA/CA ad-hoc networks," *In Proceedings of the 6th ACM symposium on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks (PE-WASUN '09).* ACM, New York, NY, USA, 33-40. 2009

[4] D. Stoyan, W. Kendall, and J. Mecke, *Stochastic Geometry and Its Applications, 2nd Edition.* John Wiley ans Sons Ltd, Chichester, UK, 1996

[5] A. Busson, "An overview of results on ad hoc network performances using spatial model" *IEEE Symposium, Computers and Communications*, vol., no., pp.36-41, 5-8 July 2009 doi: 10.1109/ISCC.2009.5202409

[6] D. Tse and P. Viswanath, *Fundamentals of wireless communication.* Cambridge University Press, 2005

[7] M. Veillette. Stable distribution Matlab package. Boston university. Available on http://math.bu.edu/people/mveillet/

[8] F. Baccelli and B. Bartek, "Stochastic Geometry and Wireless Networks Volume 1: Theory," *Foundations and Trends in Networking*, vol. 3, no. 3-4, pp. 249-449, 2009

[9] E.S. Sousa and J.A. Silvester, "Optimum transmission ranges in a direct-sequence spread spectrum multihop packet radio network", *IEEE Journal on Selected Areas in Commun.*, Vol. 8, No. 5, pp. 762-771, 1990

[10] X. Yang and N. Vaidya, "On physical carrier sensing in wireless ad hoc networks", in *IEEE Proc. Computer Commun.*, 2005

[11] T.S. Kim, H. Lim, and J.C. Hou.; , "Improving spatial reuse through tuning transmit power, carrier sense threshold, and data rate in multihop wireless networks", *In Proceedings of the 12th annual international conference on Mobile computing and networking ACM*, New York, NY, USA, 366-377. 2006

[12] I. Palasti, "On some random space filling problem". *Mathematical Institute of Hungarian Academy of Sciences*, 5(1):353.359, 1960

[13] F. Baccelli and B. Blaszczyszyn; "On a coverage process ranging from the boolean model to the poisson voronoi tessellation, with applications to wireless communications." *Advances in Applied Probability*, 33:293-323, 2001

[14] F. Baccelli, B. Blaszczyszyn, and F. Tournois; , "Spatial averages of coverage characteristics in large cdma networks". *Wireless Networks*, 8(6):569-586, 2002

[15] O. Dousse, F. Baccelli, and P. Thiran, "Impact of interferences on connectivity in ad hoc networks," *IEEE/ACM Trans. Networking*, vol.13, no.2, pp. 425- 436, April 2005

[16] J. Ilow and D. Hatzinakos, "Analytic alpha-stable noise modeling in a Poisson field of interferers or scatterers," *IEEE Trans. Signal Processing*, vol.46, no.6, pp.1601-1611, June 1998

[17] X. Yang and A.P. Petropulu, "Co-channel interference modeling and analysis in a Poisson field of interferers in wireless communications," *IEEE Trans. Signal Processing*, vol.51, no.1, pp. 64- 76, January 2003

[18] P. Cardieri, "Modeling Interference in Wireless Ad Hoc Networks," *IEEE Communications Surveys & Tutorials*, vol.12, no.4, pp.551-572, Fourth Quarter 2010

[19] J. Venkataraman, M. Haenggi, and O. Collins, "Shot Noise Models for Outage and Throughput Analyses in Wireless Ad Hoc Networks," *MILCOM*, pp. 1-7, 2006

[20] A. Busson, G. Chelius, and J.-M. Gorce, "Interference modeling in CSMA multi-hop wireless networks," *INRIA*, Research Report 6624.

[21] M. Haenggi, J.G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, "Stochastic geometry and random graphs for the analysis and design of wireless networks". *IEEE J.Sel. A. Commun.* 27, 1029-1046, September 2009

[22] A.O. Allen, *Probability, Statistics, And Queueing Theory with Computer Science Applications.* Academic Press, 1978

[23] L. Le Cam, "The central limit theorem around 1935," *Stat. Sci.*, vol.1, no.1, pp. 78-91, February 1986

[24] "Network Simulator ns-2," http://www.isi.edu/nsnam/ns/.

[25] "IEEE Std. 802.11-2007, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications," *IEEE Std.* 802.11, 2007

[26] Q. Chen, F. Schmidt-Eisenlohr, D.Jiang, M. Torrent-Moreno, L. Delgrossi, and H. Hartenstein, "Overhaul of ieee 802.11 modeling and simulation in ns-2", *Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, Chania, Crete Island, Greece, October 22-26, 2007

[27] J. Hwang and S.L. Kim, "Cross-Layer Optimization and Network Coding in CSMA/CA-Based Wireless Multihop Networks," *Trans. on Netw.*, [Online]http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5675760, DOI 10.1109/TNET.2010.2096430

[28] F. Baccelli and B. Bartek, "Stochastic Geometry and Wireless Networks Volume 2: Applications," *Foundations and Trends in Networking*, vol. 4, no. 1-2, pp. 1-312, 2009

[29] Kevin Fall and Kannan Varadhan, "The ns Manual (formerly ns Notes and Documentation)", March 2007

[30] J. S. Evans and D. Everitt, "On the teletraffic capacity of CDMA cellular networks," *IEEE Trans. Veh. Technol.*, vol. 48, no. 1, pp. 153-165, January 1999

[31] M. Z. Win, P. C. Pinto, and L. A. Shepp; , "A mathematical theory of network interference and its applications," *Proc. IEEE*, vol. 97, no. 2, pp. 205-230, February 2009

[32] A. Ghasemi and E. S. Sousa, "Interference aggregation in spectrum-sensing cognitive wireless networks," *IEEE J. Sel. Topics Signal Process.*, vol. 2, no. 1, pp. 41-56, February 2008

[33] M. Aljuaid and H. Yanikomeroglu, "Investigating the Gaussian Convergence of the Distribution of the Aggregate Interference Power in Large Wireless Networks," *Vehicular Technology, IEEE Transactions* on , vol.59, no.9, pp.4418-4424, November 2010

[34] G. Bianchi, "Performance analysis of the ieee 802.11 distributed coordination function," *IEEE Journal, Selected Areas in Communications,* vol. 18, no. 3, pp. 535-547, 2000

[35] H. Wu, Y. Peng, K. Long, and J. Ma; , "Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis and Enhancement," in *Proceeding of the IEEE Conference on Computer Communications (INFOCOM)*, New York, NY, June 2002

Appendix A

Matlab Codes

A.1 Main.m

This file illustrates main process.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%      Main PGM      %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all
close all
clc
max_iter=50000;
node= [16 158 1571];
checkbox_iteration = 100;

r=20;
R=100;
mu =1000;
P=0.001; % W

seq_total=['R' int2str(R) '-' 'r' int2str(r) '-' 'n' int2str(node(1)) '
   -' int2str(node(2)) '-'  int2str(node(3)) '-Density-'];
for k=1:size(node,2)
    n= node(k);
    density = n/(pi*100^2);
    lambda= n/(pi*(R^2));
    lambda_new      = (1-exp(-lambda*pi*r^2))/(pi*r^2);
    ppp_new_n_ceil  = ceil(lambda_new * (pi*(R^2)));
    ppp_new_n_floor = floor(lambda_new * (pi*(R^2)));
    seq=['R'int2str(R) '-' 'r'int2str(r) '-' 'n'int2str(n) '-Density-']

    %%%%%%%%%%%%%%%% Read and save Empirical result  %%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%          NS2                    %%%%%%%%%%%%%%%%
    filename0 = [ 'Den-' int2str(n) '-' int2str(r) '-SINR.dat'];
    fid0 = fopen(filename0);
```

```
Total_interference_data = textscan(fid0, '%f');
fclose(fid0);
Total_interference = Total_interference_data{1};
sort_inf=sort(Total_interference,'ascend');
index = find(sort_inf≤0.000);
interf= (sort_inf(size(index)+1:end )*(10^(-12)));
min_x = log10(min(interf));
max_x = log10(max(interf));
interval =  min_x:0.05:max_x;
x= 10.^interval;


%%%%%%%%%%%%%%%%%%%  Theoretical closed form     %%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%  PPP & Modified_PPP          %%%%%%%%%%%%%%%
% PPP
ppp_theo=erfc((lambda * pi^2) ./ (4 * sqrt(mu * ppp_theo_x)));
ppp_theo_cdf_hist_percent = ppp_theo;
ppp_theo(2:end) = ppp_theo(2:end) - ppp_theo(1:end-1);
ppp_theo_pdf_hist_percent = ppp_theo;
% PPPmd
ppp_modified_theo=erfc((lambda_new * pi^2) ./ ...
    (4 * sqrt(mu * ppp_theo_x)));
ppp_modified_theo_cdf_hist_percent = ppp_modified_theo;
ppp_modified_theo(2:end) = ppp_modified_theo(2:end) - ...
    ppp_modified_theo(1:end-1);
ppp_modified_theo_pdf_hist_percent = ppp_modified_theo;


%%%%%%%%%%%%%%%%%%%        Simulation                %%%%%%%
%%%%%%%%%%%%%%%%%%%   PPP & PPPmd & Matern & SSI      %%%%%%%
ppp_interf = [];
ppp_new_interf_ceil = [];
ppp_new_interf_floor = [];
matern_interf = [];
ssi_interf = [];
for a=1:max_iter

    %%%%%%%%%%%%%%%%%%  Node distribution  %%%%%%%%%%%%%%%%%%
    % PPP
    ppp_theta=rand(1,n)*2*pi;
    ppp_dist=R*sqrt(rand(1,n));
    ppp_mother_node=[ppp_theta;ppp_dist];
    % PPPmd_ceil
    ppp_new_theta_ceil=rand(1,ppp_new_n_ceil)*2*pi;
    ppp_new_dist_ceil=R*sqrt(rand(1,ppp_new_n_ceil));
    ppp_new_mother_node_ceil= ...
                      [ppp_new_theta_ceil;ppp_new_dist_ceil];
    % PPPmd_floor
    ppp_new_theta_floor=rand(1,ppp_new_n_floor)*2*pi;
    ppp_new_dist_floor=R*sqrt(rand(1,ppp_new_n_floor));
    ppp_new_mother_node_floor= ...
                    [ppp_new_theta_floor;ppp_new_dist_floor];
    % Matern
    theta=rand(1,n)*2*pi;
```

```
dist=R*sqrt(rand(1,n));
mother_node=[theta;dist];
matern_selec=mother_node(:,1);
matern_counter_node=mother_node(:,1);
for i=2:n
    matern_fail=sum(sqrt(dist(i)^2+ ...
                matern_counter_node(2,:).^2- ...
        2*dist(i).*matern_counter_node(2,:).* ...
        cos(theta(i)-matern_counter_node(1,:)))<r);
    if matern_fail==0;
        matern_selec=[matern_selec,mother_node(:,i)];
    end
    matern_counter_node= ...
              [matern_counter_node,mother_node(:,i)];
end
%SSI
ssi_selec=mother_node(:,1);
for i=2:1:n
    ssi_fail=sum(sqrt(dist(i)^2+ssi_selec(2,:).^2- ...
             2*dist(i).*ssi_selec(2,:).* ...
             cos(theta(i)-ssi_selec(1,:)))<r);
    if ssi_fail==0;
        ssi_selec=[ssi_selec,mother_node(:,i)];
    end
end

%%%%%%%%%%%%%%%%%%  Channel model  %%%%%%%%%%%%%%%%%%
% PPP
ppp_X=-log(1-rand(1,size(ppp_mother_node,2)));
ppp_l=P*ppp_X.*min(1,ppp_mother_node(2,:).^(-4));
ppp_interf = [ppp_interf sum(ppp_l)];
% PPPmd_ceil
ppp_new_X_ceil= ...
-log(1-rand(1,size(ppp_new_mother_node_ceil,2)));
ppp_new_l_ceil= ...
P*ppp_new_X_ceil.*min(1,ppp_new_mother_node_ceil(2,:).^(-4));
ppp_new_interf_ceil = ...
[ppp_new_interf_ceil sum(ppp_new_l_ceil)];
% PPPmd_floor
ppp_new_X_floor= ...
-log(1-rand(1,size(ppp_new_mother_node_floor,2)));
ppp_new_l_floor=         P*ppp_new_X_floor.*min(1,
   ppp_new_mother_node_floor(2,:).^(-4));
ppp_new_interf_floor = [ppp_new_interf_floor sum(
   ppp_new_l_floor)];
% Matern
matern_X=-log(1-rand(1,size(matern_selec,2)));
matern_l=P*matern_X.*min(1,matern_selec(2,:).^(-4));
matern_interf = [matern_interf sum(matern_l)];
% SSI
ssi_X=-log(1-rand(1,size(ssi_selec,2)));
ssi_l=P*ssi_X.*min(1,ssi_selec(2,:).^(-4));
ssi_interf = [ssi_interf sum(ssi_l)];
```

```
    %%%%%%%%%%%%%%%%%%  Erase memory   %%%%%%%%%%%%%%%%%%
    clear ppp_theta ppp_dist ppp_mother_node;
    clear ppp_new_theta ppp_new_dist;
    clear ppp_new_mother_node ppp_new_theta_floor;
    clear ppp_new_dist_floor;
    clear ppp_new_mother_node_floor ppp_X ppp_l;
    clear ppp_new_X ppp_new_l;
    clear ppp_new_X_floor ppp_new_l_floor;
    clear theta dist mother_node i;
    clear matern_selec clear matern_fail;
    clear matern_X matern_l;
    clear ssi_selec ssi_fail ssi_X ssi_l;
end


%%%%%%%%%%%%%%%%%%%        PDF and CDF                 %%%%
%%%%%%%%%%%%%%%%%%%   NS2 & PPP & PPPmd & Matern & SSI %%%%
% NS2
pdf_hist = histc(interf,x);
pdf_hist_sum = max(cumsum(pdf_hist));
pdf_hist_percent = pdf_hist        / pdf_hist_sum;
cdf_hist_percent =  (cumsum(pdf_hist)/ pdf_hist_sum);
ppp_theo_interval =  interval;
ppp_theo_x= x;
extrapolation('ns2',interf, R, r, n,max_iter );
% PPP
ppp_pdf_hist = histc(ppp_interf,x);
ppp_pdf_hist_cumsum = cumsum(ppp_pdf_hist);
ppp_pdf_hist_sum = max(ppp_pdf_hist_cumsum);
ppp_pdf_hist_percent = ppp_pdf_hist/ppp_pdf_hist_sum;
ppp_cdf_hist = cumsum(ppp_pdf_hist);
ppp_cdf_hist_max = max(ppp_cdf_hist);
ppp_cdf_hist_percent =  (ppp_cdf_hist/ppp_cdf_hist_max);
% PPPmd_ceil
ppp_new_pdf_hist_ceil = histc(ppp_new_interf_ceil,x);
ppp_new_pdf_hist_cumsum_ceil = cumsum(ppp_new_pdf_hist_ceil);
ppp_new_pdf_hist_sum_ceil = max(ppp_new_pdf_hist_cumsum_ceil);
ppp_new_pdf_hist_percent_ceil = ppp_new_pdf_hist_ceil/
   ppp_new_pdf_hist_sum_ceil;
ppp_new_cdf_hist_ceil = cumsum(ppp_new_pdf_hist_ceil);
ppp_new_cdf_hist_max_ceil = max(ppp_new_cdf_hist_ceil);
ppp_new_cdf_hist_percent_ceil = (ppp_new_cdf_hist_ceil/
   ppp_new_cdf_hist_max_ceil);
% PPPmd_floor
ppp_new_pdf_hist_floor = histc(ppp_new_interf_floor,x);
ppp_new_pdf_hist_cumsum_floor = cumsum(ppp_new_pdf_hist_floor);
ppp_new_pdf_hist_sum_floor = max(ppp_new_pdf_hist_cumsum_floor);
ppp_new_pdf_hist_percent_floor = ppp_new_pdf_hist_floor/
   ppp_new_pdf_hist_sum_floor;
ppp_new_cdf_hist_floor = cumsum(ppp_new_pdf_hist_floor);
ppp_new_cdf_hist_max_floor = max(ppp_new_cdf_hist_floor);
ppp_new_cdf_hist_percent_floor = ppp_new_cdf_hist_floor/
   ppp_new_cdf_hist_max_floor;
% matern
```

```
matern_pdf_hist = histc(matern_interf,x);
matern_pdf_hist_cumsum = cumsum(matern_pdf_hist);
matern_pdf_hist_sum = max(matern_pdf_hist_cumsum);
matern_pdf_hist_percent = matern_pdf_hist/matern_pdf_hist_sum;
matern_cdf_hist = cumsum(matern_pdf_hist);
matern_cdf_hist_max = max(matern_cdf_hist);
matern_cdf_hist_percent = (matern_cdf_hist/matern_cdf_hist_max);
% ssi
ssi_pdf_hist = histc(ssi_interf,x);
ssi_pdf_hist_cumsum = cumsum(ssi_pdf_hist);
ssi_pdf_hist_sum = max(ssi_pdf_hist_cumsum);
ssi_pdf_hist_percent = ssi_pdf_hist/ssi_pdf_hist_sum;
ssi_cdf_hist = cumsum(ssi_pdf_hist);
ssi_cdf_hist_max = max(ssi_cdf_hist);
ssi_cdf_hist_percent = (ssi_cdf_hist/ssi_cdf_hist_max);


%%%%%%%%%%%%%%%      Kolmogorov-Smirnov Test            %%%%%
%%%%%%%%%%%%%%%  NS2 vs (PPP || PPPmd || Matern || SSI) %%%%%
[h1,p1,k1]=kstest2(interf, ppp_interf);
[h2,p2,k2]=kstest2(interf, ppp_new_interf_ceil);
[h3,p3,k3]=kstest2(interf, matern_interf);
[h4,p4,k4]=kstest2(interf, ssi_interf);
PP = ['R' int2str(R) '-r' int2str(r) '-n' int2str(n) '-Static-
   kstest'];
filename0=[ PP '-ns2-process.txt'];
fid = fopen(filename0, 'wt');
fprintf(fid, filename0);
fprintf(fid, '\nns2 vs ppp     h=  %d\n', h1);
fprintf(fid, 'ns2 vs ppp     p=  %d\n', p1);
fprintf(fid, 'ns2 vs ppp     k=  %d\n', k1);
fprintf(fid, 'ns2 vs cm      h=  %d\n', h2);
fprintf(fid, 'ns2 vs cm      p=  %d\n', p2);
fprintf(fid, 'ns2 vs cm      k=  %d\n', k2);
fprintf(fid, 'ns2 vs tm      h=  %d\n', h3);
fprintf(fid, 'ns2 vs tm      p=  %d\n', p3);
fprintf(fid, 'ns2 vs tm      k=  %d\n', k3);
fprintf(fid, 'ns2 vs ssi     h=  %d\n', h4);
fprintf(fid, 'ns2 vs ssi     p=  %d\n', p4);
fprintf(fid, 'ns2 vs ssi     k=  %d\n', k4);
fclose(fid);


%%%%%%%%%%%%%%%        Extrapolation process            %%%%%
%%%%%%%%%%%%%%%     NS2 vs heavy tailed distribution    %%%%%
extrapolation(checkbox_iteration, interf,max_iter,R, r, n);


%%%%%%%%%%%%%%%        Figure generation                %%%%%
%%%%%%%%%%%%%%%          PDF & CDF                      %%%%%
% Set filename to save figures into *.fig files
filename1=[ seq 'PDF-Normscale-Process' ];
filename2=[ seq 'CDF-Logscale-Process'  ];
filename3=[ seq 'PDF-Normscale-Process-ns2'  ];
```

```
filename4=[ seq 'CDF-Logscale-Process-ns2' ];
filename5=[ seq_total 'PDF-Normscale'  ];
filename6=[ seq_total 'CDF-Logscale'  ];
fignumber1= (n)+1;
fignumber2= (n)+2;
fignumber3= (n)+3;
fignumber4= (n)+4;

% PDF-Normscale for Comparison among Processes)
fig1=figure(fignumber1);
plot(x,ppp_pdf_hist_percent,'--','Color',[0 0 1],'LineWidth',2);
hold on;
plot(x,ppp_new_pdf_hist_percent_ceil,'-' ,'Color',[0 .5 0],'
   LineWidth',2);
hold on;
plot(x,matern_pdf_hist_percent,'-.','Color',[1 0 0],'LineWidth',2);
hold on;
plot(x,ssi_pdf_hist_percent,':' ,'Color',[0 0 0] ,'LineWidth',2);
legend('PPP','PPPmd','MPP','SSI','Location','NorthEast');
grid off;
xlim([min(x) (1*10^(-8))]);
xlabel('Power(W) on Normal Scale');
ylabel('Probability Density Function');
saveas(fig1,filename1,'fig');

% CDF-Logscale for Comparison among Processes)
fig2=figure(fignumber2);
semilogx(x,ppp_cdf_hist_percent ,'--' ,'Color',[0 0 1],'LineWidth'
   ,2);
hold on;
semilogx(x,ppp_new_cdf_hist_percent_ceil,'-' ,'Color',[0 .5 0],'
   LineWidth',2);
hold on;
semilogx(x,matern_cdf_hist_percent,'-.','Color',[1 0 0],'LineWidth'
   ,2);
hold on;
semilogx(x,ssi_cdf_hist_percent,':' ,'Color',[0 0 0] ,'LineWidth'
   ,2);
legend('PPP','PPPmd','MPP','SSI','Location','NorthWest');
xlim([10^(-11) 10^(-4)]);
xlabel('Power(W) on Log Scale');
ylabel('Cummulative Distribution Function');
grid off;
saveas(fig2,filename2,'fig');

% PDF-Normscale for Comparison among NS2 and Processes
fig3=figure(fignumber3);
plot(x,pdf_hist_percent ,'o' ,'Color',[0 0 0],'LineWidth',2);
hold on;
plot(x,ppp_pdf_hist_percent,'--' ,'Color',[0 0 1],'LineWidth',2);
hold on;
plot(x,ppp_new_pdf_hist_percent_ceil,'-' ,'Color',[0 .5 0],'
   LineWidth',2);
hold on;
```

```
plot(x,matern_pdf_hist_percent,'-.','Color',[1 0 0],'LineWidth',2);
hold on;
plot(x,ssi_pdf_hist_percent,':' ,'Color',[0 0 0] ,'LineWidth',2);
legend('NS2','PPP','PPPmd','MPP','SSI','Location','NorthEast');
grid off;
xlim([min(x) (1*10^(-8))]);
xlabel('Power(W) on Normal Scale');
ylabel('Probability Density Function');
saveas(fig3,filename3,'fig');

% CDF-Logscale for Comparison among NS2 and Processes
fig4=figure(fignumber4);
semilogx(x,cdf_hist_percent,'o' ,'Color',[0 0 0],'LineWidth',2);
hold on;
semilogx(x,ppp_cdf_hist_percent,'--' ,'Color',[0 0 1],'LineWidth'
    ,2);
hold on;
semilogx(x,ppp_new_cdf_hist_percent_ceil,'-' ,'Color',[0 .5 0],'
    LineWidth',2);
hold on;
semilogx(x,matern_cdf_hist_percent,'-.','Color',[1 0 0],'LineWidth'
    ,2);
hold on;
semilogx(x,ssi_cdf_hist_percent,':' ,'Color',[0 0 0] ,'LineWidth'
    ,2);
legend('NS2','PPP','PPPmd','MPP','SSI','Location','NorthEast');
xlim([10^(-11) 10^(-4)]);
xlabel('Power(W) on Log Scale');
ylabel('Cummulative Distribution Function');
grid off;
saveas(fig4,filename4,'fig');

% PDF-Normcale for Different densities
fig5=figure(5);
if k==1
    plot(x,pdf_hist_percent,'--' ,'Color',[0 0 1],'LineWidth',2);
    hold on;
elseif k==2
    plot(x,pdf_hist_percent,'-' ,'Color',[1 0 0],'LineWidth',2);
    hold on;
else
    plot(x,pdf_hist_percent,':' ,'Color',[0 0 0],'LineWidth',2);
    hold on;
end
legend('\lambda=0.0005','\lambda=0.005','\lambda=0.05' );
xlim([min(x) (1*10^(-9))]);
xlabel('Power(W) on Normal Scale');
ylabel('Probability Density Function');
grid off;
saveas(fig5,filename5,'fig');

% CDF-Normcale for Different densities
fig6=figure(6);
if k==1
```

```
            semilogx(x,cdf_hist_percent,'--' ,'Color',[0 0 1],'LineWidth'
                ,2);
            hold on;
        elseif k==2
            semilogx(x,cdf_hist_percent,'-' ,'Color',[1 0 0],'LineWidth',2)
                ;
            hold on;
        else
            semilogx(x,cdf_hist_percent,':' ,'Color',[0 0 0],'LineWidth',2)
                ;
            hold on;
        end
        xlim([10^(-13) 10^(-4)]);
        legend('\lambda=0.0005','\lambda=0.005','\lambda=0.05' );
        xlabel('Power(W) on Log Scale');
        ylabel('Cummulative Distribution Function');
        grid off;
        saveas(fig6,filename6,'fig');
    end % End for k=1:size(node,2)
```

A.2 Extrapolation2.m

This file illustrates extrapolation process.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%     Extrapolation Process      %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function extrapolation(checkbox_iteration, long_interf, max_iter,R, r,
   n)

epsilon = 1e-20;

%%%%%%%%%%%%%%%%%%%        Check box generation             %%%%%%
%%%%%%%%%%%%%%%%%         heavy tailed distribution         %%%%%%
for j=1 : checkbox_iteration

    % Sampling 50000*2
    sampling_index = randsample(length(long_interf), max_iter*2);
    interf         = long_interf(sampling_index(1:max_iter));
    interf_fitting = long_interf(sampling_index(max_iter+1:end));

    % Estimate parameters for each distribution with raw data
    wbl(j,:)    =wblfit(interf_fitting);
    p(j,:)      = stblfit(interf_fitting,'ecf',statset('Display','iter'
       ))';
    parmhat(j,:)= lognfit(interf_fitting);
    [muhat,sigmahat] = normfit(interf_fitting);

    % PDF for Alpha-stable fit
    alpha_x1    = stblcdf(interf + epsilon, p(j,1),p(j,2),p(j,3),p(j,4)
       );
    alpha_x2    = stblcdf(interf - epsilon, p(j,1),p(j,2),p(j,3),p(j,4)
       );
    probs_alpha = max(alpha_x1 - alpha_x2, 0);

    % PDF for Log-normal fit
    logn_x1     = logncdf(interf+ epsilon ,parmhat(j,1),parmhat(j,2));
    logn_x2     = logncdf(interf- epsilon ,parmhat(j,1),parmhat(j,2));
    probs_logn  = max(logn_x1-logn_x2, 0);

    % PDF for Weibull fit
    wbl_x1      =  wblcdf(interf+ epsilon, wbl(j,1),wbl(j,2));
    wbl_x2      =  wblcdf(interf- epsilon, wbl(j,1),wbl(j,2));
    probs_wbl   = max(wbl_x1- wbl_x2, 0);
    probs_alpha( probs_alpha == 0)  = 1e-20;
    probs_logn( probs_logn == 0)    = 1e-20;
    probs_wbl( probs_wbl == 0)      = 1e-20;

    % Summation to avoid numerical overflows
    Prob_ML_alpha(j,1)  =(sum(log(probs_alpha)));
    Prob_ML_logn(j,1)   =(sum(log(probs_logn)));
    Prob_ML_wbl(j,1)    =(sum(log(probs_wbl)));
    % Save the variables
    savevars.wbl(j,:)       = wbl(j,:);
    savevars.p(j,:)         = p(j,:);
    savevars.parmhat(j,:)   = parmhat(j,:);
    savevars.Prob_alpha(j)  = Prob_ML_alpha(j);
    savevars.Prob_logn(j)   = Prob_ML_logn(j);
    savevars.Prob_wbl(j)    = Prob_ML_wbl(j);
end

%%%%%%%%%%%%%%%%%%%            Chi-square2                  %%%%%%
%%%%%%%%%%%%%%%%%         heavy tailed distribution         %%%%%%

% Set for Chi-square goodness of fit test
X = sort(interf, 'descend');
n_samples = size(X,1);
max_bins = 10;
Alpha_min_bin_items = max_iter/max_bins;
Log_min_bin_items   = max_iter/max_bins;
Wbl_min_bin_items   = max_iter/max_bins;

% Alpha-stable
Alpha_cdf_func = {@stblcdf, p(j,1),p(j,2),p(j,3),p(j,4)};
Alpha_adj_edges = [];
Alpha_curr_items = [];
Alpha_bin_sz = [];
Alpha_cdf_bin_sz = [];
for i=1:n_samples
   Alpha_curr_items = [Alpha_curr_items X(i)];
   if (length(Alpha_curr_items) ≥ Alpha_min_bin_items)
       Alpha_cdf_item_count = n_samples * ...
       Alpha_cdf_func{1}(Alpha_curr_items(1), Alpha_cdf_func{2:end}) -
          ...
       Alpha_cdf_func{1}(Alpha_curr_items(end), Alpha_cdf_func{2:end}))
          ;
       if (Alpha_cdf_item_count > 5)
         Alpha_adj_edges = [(Alpha_curr_items(end) - 1e-50)
            Alpha_adj_edges ];
         Alpha_bin_sz = [length(Alpha_curr_items) Alpha_bin_sz];
         Alpha_cdf_bin_sz = [Alpha_cdf_item_count Alpha_cdf_bin_sz];
         Alpha_curr_items = [];
         if (length(Alpha_adj_edges) ≥ max_bins - 1)
             break;
         end
       end
   end
   if (mod(i, 50000) == 0)
       fprintf(2, '%d\n', i);
   end
end
Alpha_adj_edges1 = [min(X) Alpha_adj_edges max(X)];
[h1,p1,st1]=chi2gof(interf,'cdf',Alpha_cdf_func, 'edges',
   Alpha_adj_edges1);

% Log-normal
```

```
Log_cdf_func = {@logncdf, parmhat(j,1), parmhat(j,2)};
Log_adj_edges = [];
Log_curr_items = [];
Log_bin_sz = [];
Log_cdf_bin_sz = [];
for i=1:n_samples
   Log_curr_items = [Log_curr_items X(i)];
   if (length(Log_curr_items) ≥ Log_min_bin_items)
       Log_cdf_item_count = n_samples * ...
           (Log_cdf_func{1}(Log_curr_items(1), Log_cdf_func{2:end}) -
              ...
             Log_cdf_func{1}(Log_curr_items(end), Log_cdf_func{2:end}))
                ;
       if (Log_cdf_item_count > 5)
         Log_adj_edges = [(Log_curr_items(end) - 1e-50) Log_adj_edges
            ];
         Log_bin_sz = [length(Log_curr_items) Log_bin_sz];
         Log_cdf_bin_sz = [Log_cdf_item_count Log_cdf_bin_sz];
         Log_curr_items = [];

         if (length(Log_adj_edges) ≥ max_bins - 1)
             break;
         end
       end
   end
   if (mod(i, 50000) == 0)
       fprintf(2, '%d\n', i);
   end
end
Log_adj_edges1 = [min(X) Log_adj_edges max(X)];
[h2,p2,st2]=chi2gof(interf,'cdf',Log_cdf_func, 'edges', Log_adj_edges1)
   ;

% Weibull
Wbl_cdf_func = {@wblcdf, wbl(j,1),wbl(j,2)};
Wbl_adj_edges = [];
Wbl_curr_items = [];
Wbl_bin_sz = [];
Wbl_cdf_bin_sz = [];
for i=1:n_samples
   Wbl_curr_items = [Wbl_curr_items X(i)];
   if (length(Wbl_curr_items) ≥ Wbl_min_bin_items)
       Wbl_cdf_item_count = n_samples * ...
           (Wbl_cdf_func{1}(Wbl_curr_items(1), Wbl_cdf_func{2:end}) -
              ...
             Wbl_cdf_func{1}(Wbl_curr_items(end), Wbl_cdf_func{2:end}))
                ;
       if (Wbl_cdf_item_count > 5)
         Wbl_adj_edges = [(Wbl_curr_items(end) - 1e-50) Wbl_adj_edges
            ];
         Wbl_bin_sz = [length(Wbl_curr_items) Wbl_bin_sz];
         Wbl_cdf_bin_sz = [Wbl_cdf_item_count Wbl_cdf_bin_sz];
         Wbl_curr_items = [];
         if (length(Wbl_adj_edges) ≥ max_bins - 1)
             break;
         end
       end
   end
   if (mod(i, 50000) == 0)
       fprintf(2, '%d\n', i);
   end
end
Wbl_adj_edges1 = [min(X) Wbl_adj_edges max(X)];
[h3,p3,st3]=chi2gof(interf,'cdf',Wbl_cdf_func, 'edges', Wbl_adj_edges1)
    ;

%%%%%%%%%%%%%%%%%          Kolmogorov-Smirnov Test          %%%%%%
%%%%%%%%%%%%%%%%%         heavy tailed distribution         %%%%%%

min_x = log10(min(X));
max_x = log10(max(X));
interval =  min_x:0.1:max_x;
x= 10.^interval;

% NS2
pdf_hist = histc(interf,x);
pdf_hist_cumsum = cumsum(pdf_hist);
pdf_hist_sum = max(pdf_hist_cumsum);
pdf_hist_percent = pdf_hist/pdf_hist_sum;
cdf_hist = cumsum(pdf_hist);
cdf_hist_max = max(cdf_hist);
cdf_hist_percent = cdf_hist/cdf_hist_max;

% Alpha Stable
alphacdf_x=stblcdf(x,p(j,1),p(j,2),p(j,3),p(j,4));
alphapdf_x=zeros(1,size(x,2));
alphapdf_x(2:end) = alphacdf_x(2:end) - alphacdf_x(1:end-1);
uniform_alpha_inverse = alphacdf_x(1) + ...
                       (alphacdf_x(end) - alphacdf_x(1)).*rand(max_iter
                          ,1);
sample_alpha_cdf =  sort(uniform_alpha_inverse, 'ascend');
sample_alpha = stblinv(sample_alpha_cdf,p(j,1),p(j,2),p(j,3),p(j,4));
[h22,p22,k22]=kstest2(interf, sample_alpha);

% Lognormal
logncdf_x = logncdf(x,parmhat(j,1),(parmhat(j,2)));
lognpdf_x = zeros(1,size(x,2));
lognpdf_x(2:end) = logncdf_x(2:end) - logncdf_x(1:end-1);
uniform_logn_inverse = logncdf_x(1) + ...
                         (logncdf_x(end) - logncdf_x(1)).*rand(max_iter
                            ,1);
sample_logn_cdf =  sort(uniform_logn_inverse, 'ascend');
sample_logn = logninv(sample_logn_cdf,parmhat(j,1),(parmhat(j,2)));
[h33,p33,k33]=kstest2(interf, sample_logn);

% Weibull
weicdf_x = wblcdf(x,wbl(j,1),wbl(j,2));
```

```
weipdf_x = zeros(1,size(x,2));
weipdf_x(2:end) = weicdf_x(2:end) - weicdf_x(1:end-1);
uniform_wei_inverse = weicdf_x(1) + ...
                           (weicdf_x(end) - weicdf_x(1)).*rand(max_iter
                              ,1);
sample_wei_cdf =  sort(uniform_wei_inverse, 'ascend');
sample_wei = wblinv(sample_wei_cdf,wbl(j,1),wbl(j,2));
[h44,p44,k44]=kstest2(interf, sample_wei);


%%%%%%%%%%%%%%%           Figure generation                 %%%%%%
%%%%%%%%%%%%%%%         Box plot and histogram              %%%%%%

% Box Plot Generation
PP3 = ['R' int2str(R) '-' 'r' int2str(r) '-' 'n' int2str(n) ];
filename3=[ PP3 '-logarithmic-Boxplot'];
fig1=figure(1);
boxplot([Prob_ML_alpha Prob_ML_logn Prob_ML_wbl], 'labels', ...
                                {'Alpha-stable', 'Log-normal', 'Weibull
                                   '});
ylabel('Log-probability');
saveas(fig1,filename3,'fig');

% NS2 raw data bar plot
X = interf_fitting;
min_x = log10(min(X));
max_x = log10(max(X));
interval =  min_x:0.05:max_x;
x= 10.^interval;
x2 = zeros(1,size(x,2)*4-1);
x2(1) = x(1);
x3 = x(1);
for l=2:size(x,2)
    x2(4*(l-1)) = (x(l) + x(l-1))/2;
    x2(4*(l-1)+1) = (x(l) + x(l-1))/2;
    x2(4*(l-1)+2) = x(l);
    x2(4*(l-1)+3) = x(l);
    x3 = [x3 x(l)];
end
x3(end) = [];
x3(1) = [];
pdf_hist = histc(X,x);
pdf_hist_cumsum = cumsum(pdf_hist);
pdf_hist_sum = max(pdf_hist_cumsum);
pdf_hist_percent = pdf_hist/pdf_hist_sum;
cdf_hist = cumsum(pdf_hist);
cdf_hist_max = max(cdf_hist);
cdf_hist_percent = cdf_hist/cdf_hist_max;
pdf_hist2 = zeros(size(pdf_hist_percent,1)*4-1,1);
pdf_hist3 = [];
for l=1:size(pdf_hist_percent,1)-1
    pdf_hist2(4*(l-1)+1) = pdf_hist_percent(l);
    pdf_hist2(4*(l-1)+2) = pdf_hist_percent(l);
    pdf_hist2(4*(l-1)+3) = pdf_hist_percent(l+1);
    pdf_hist2(4*(l-1)+4) = pdf_hist_percent(l+1);
    pdf_hist3 = [pdf_hist3 pdf_hist_percent(l)];
end
pdf_hist3(1) =[];

% Alpha stable fit
alphacdf_x=stblcdf(x,p(j,1),p(j,2),p(j,3),p(j,4));
alphapdf_x=zeros(1,size(x,2));
alphapdf_x(2:end) = alphacdf_x(2:end) - alphacdf_x(1:end-1);
% Lognormal fit
logncdf_x = logncdf(x,parmhat(j,1),parmhat(j,2));
lognpdf_x = zeros(1,size(x,2));
lognpdf_x(2:end) = logncdf_x(2:end) - logncdf_x(1:end-1);
% Weibull fit
weicdf_x = wblcdf(x,wbl(j,1),wbl(j,2));
weipdf_x = zeros(1,size(x,2));
weipdf_x(2:end) = weicdf_x(2:end) - weicdf_x(1:end-1);

% Plot Generation
filename10 = [ PP3 '-plot-Norm-ns2-alpha-logn-weibull'];
filename20 = [ PP3 '-plot-Log-ns2-alpha-logn-weibull' ];
fignumber=20+n;
fig20=figure(fignumber);
l1=semilogx(x,alphapdf_x      ,'-+' ,'Color',[0 0 1],'LineWidth',1);
   hold on;
l2=semilogx(x,lognpdf_x       ,'-*' ,'Color',[1 0 0],'LineWidth',1);
   hold on;
l3=semilogx(x,weipdf_x        ,'-x' ,'Color',[0 1 0],'LineWidth',1);
   hold on;
l4=semilogx(x2,pdf_hist2,'k-');hold on;
l4=bar(x3,pdf_hist3,0);hold on;
set(l4,'FaceColor',[1 1 1]);
legend([l1 l2 l3 l4], {'alpha-stable', 'log-normal','weibull','ns2'});
xlim([min(X)+1e-13 max(X)-1e-13]);
xlabel('Interference Power(W) on Log Scale');
ylabel('Probability Density Function');
grid off;
saveas(fig20,filename20,'fig');
hold off;

%%%%%%%%%%%%%%%           data file generation              %%%%%%
%%%%%%%%%%%%%%%          chi2test and kstest2               %%%%%%
PP2 = ['R' int2str(R) '-' 'r' int2str(r) '-' 'n' int2str(n) '-Chi2+
   KStest-'];
filename1=[ PP2 'NS2-Alpha+Logn+Wbl.txt'];
fid = fopen(filename1, 'wt');
fprintf(fid, filename1);

fprintf(fid, '\n
   ********************************************************');
fprintf(fid, '\n
   ********************************************************');
fprintf(fid, '\n
   ********************************************************');
```

```
fprintf(fid, '\n**********  NS2 and Distributions chi2test
    *************');
fprintf(fid, '\n\n***NS2 and Alpha chi2test***\n');
fprintf(fid, 'sample size   %d\n', max_iter);
fprintf(fid, 'result        %d\n', h1);
fprintf(fid, 'p-value       %d\n', p1);
fprintf(fid, 'chi2statistic %d\n', st1.chi2stat);
fprintf(fid, 'degreefreedom %d', st1.df);
fprintf(fid, '\nedges ');
fprintf(fid, '%d ', st1.edges);
fprintf(fid, '\nalpha observe ');
fprintf(fid, '%d ', st1.O);
fprintf(fid, '\nalpha estimation ');
fprintf(fid, '%d ', st1.E);
fprintf(fid, '\nalpha-stable parameters_1: %d ', p(:,1));
fprintf(fid, '\nalpha-stable parameters_2: %d ', p(:,2));
fprintf(fid, '\nalpha-stable parameters_3: %d ', p(:,3));
fprintf(fid, '\nalpha-stable parameters_4: %d ', p(:,4));

fprintf(fid, '\n\n***NS2 and Logn chi2test***\n');
fprintf(fid, 'sample size   %d\n', max_iter);
fprintf(fid, 'result        %d\n', h2);
fprintf(fid, 'p-value       %d\n', p2);
fprintf(fid, 'chi2statistic %d\n', st2.chi2stat);
fprintf(fid, 'degreefreedom %d', st2.df);
fprintf(fid, '\nedges ');
fprintf(fid, '%d ', st2.edges);
fprintf(fid, '\nLogNormal observe ');
fprintf(fid, '%d ', st2.O);
fprintf(fid, '\nLogNormal estimation ');
fprintf(fid, '%d ', st2.E);
fprintf(fid, '\nLog-normal parameters_1: %d ', parmhat(:,1));
fprintf(fid, '\nLog-normal parameters_2: %d ', parmhat(:,2));

fprintf(fid, '\n\n***NS2 and Wbl chi2test***\n');
fprintf(fid, 'sample size   %d\n', max_iter);
fprintf(fid, 'result        %d\n', h3);
fprintf(fid, 'p-value       %d\n', p3);
fprintf(fid, 'chi2statistic %d\n', st3.chi2stat);
fprintf(fid, 'degreefreedom %d', st3.df);
fprintf(fid, '\nedges ');
fprintf(fid, '%d ', st3.edges);
fprintf(fid, '\nWeibull observe ');
fprintf(fid, '%d ', st3.O);
fprintf(fid, '\nWeibull estimation ');
fprintf(fid, '%d ', st3.E);
fprintf(fid, '\nWeibull parameters: %d %d ', wbl(1),wbl(2));
fprintf(fid, '\nWeibull parameters_1: %d ', wbl(:,1));
fprintf(fid, '\nWeibull parameters_2: %d ', wbl(:,2));

fprintf(fid, '\n
    *******************************************************');
fprintf(fid, '\n
    *******************************************************');
```

```
fprintf(fid, '\n
   ********************************************************');
fprintf(fid, '\n**********  NS2 and Distributions kstest2
   **************');
fprintf(fid, '\n\n***NS2 and Alpha kstest2***\n');
fprintf(fid, 'sample size   %d\n', max_iter);
fprintf(fid, 'result        %d\n', h22);
fprintf(fid, 'p-value       %d\n', p22);
fprintf(fid, 'chi2statistic %d\n', k22);
fprintf(fid, '\n***NS2 and Logn kstest2***\n');
fprintf(fid, 'sample size   %d\n', max_iter);
fprintf(fid, 'result        %d\n', h33);
fprintf(fid, 'p-value       %d\n', p33);
fprintf(fid, 'chi2statistic %d\n', k33);
fprintf(fid, '\n***NS2 and Wbl kstest2***\n');
fprintf(fid, 'sample size   %d\n', max_iter);
fprintf(fid, 'result        %d\n', h44);
fprintf(fid, 'p-value       %d\n', p44);
fprintf(fid, 'chi2statistic %d\n', k44);
cv = 1.36*sqrt( (n_samples+n_samples) / (n_samples*n_samples) );
fprintf(fid, '\n\nCritical value %d\n', cv);

fprintf(fid, '\n
   ********************************************************');
fprintf(fid, '\n
   ********************************************************');
fprintf(fid, '\n
   ********************************************************');
fprintf(fid, '\n******** Distribution parameters in Table
   **************');
fprintf(fid, '\nalpha-stable parameters_1(ALPHA): %d ', p(:,1));
fprintf(fid, '\n');
fprintf(fid, '\nalpha-stable parameters_2(BETA): %d ', p(:,2));
fprintf(fid, '\n');
fprintf(fid, '\nalpha-stable parameters_3(GAM): %d ', p(:,3));
fprintf(fid, '\n');
fprintf(fid, '\nalpha-stable parameters_4(DELTA): %d ', p(:,4));
fprintf(fid, '\n');
fprintf(fid, '\n**********************')
fprintf(fid, '\nLog-normal parameters_1: %d ', parmhat(:,1));
fprintf(fid, '\n');
fprintf(fid, '\nLog-normal parameters_2: %d ', parmhat(:,2));
fprintf(fid, '\n**********************')
fprintf(fid, '\nWeibull parameters_1: %d ', wbl(:,1));
fprintf(fid, '\n');
fprintf(fid, '\nWeibull parameters_2: %d ', wbl(:,2));
fprintf(fid, '\n**********************')
fprintf(fid, '\nProb_ML_alpha=     %d    ', Prob_ML_alpha);
fprintf(fid, '\n');
fprintf(fid, '\nProb_ML_logn=      %d    ', Prob_ML_logn);
fprintf(fid, '\n');
fprintf(fid, '\nProb_ML_wbl =       %d    ', Prob_ML_wbl);
fprintf(fid, '\n');
fprintf(fid, '\nMed_Prob_ML_alpha=     %d\n', median(Prob_ML_alpha));
fprintf(fid, 'Med_Prob_ML_logn=      %d\n', median(Prob_ML_logn));
fprintf(fid, 'Med_Prob_ML_wbl =      %d\n', median(Prob_ML_wbl));
fprintf(fid, '\n')
fprintf(fid, '\n')
fprintf(fid, '\n')
fclose(fid);
```

Appendix B

NS2 Scripts

B.1 Main.tcl

This file illustrates main process.

```
# Main.tcl
# By Byungjin Cho

#=================
# MAC Layer Setup
#=================
set val(power)   0.001
set val(seq)     [lindex $argv 0];
set val(nn)      [lindex $argv 1];
set val(CSrange) [lindex $argv 2];
set val(Grid)    [lindex $argv 3];
set val(Scen)    [lindex $argv 4];
set val(Noise)   [expr ( ( $val(power)/($val(CSrange)*$val(CSrange)
   *$val(CSrange)*$val(CSrange))) /3.1623 ) ]
set val(nf)      [expr ( ( $val(Noise)*1e12 )  ) ]
set val(PMrange) [expr ( ( $val(Grid)*1.5 )  ) ];
set val(NFrange) [expr ( sqrt(sqrt($val(power)/$val(Noise))))]
puts "CSrange=$val(CSrange)"
puts "NFrange=$val(NFrange)"
Mac/802_11Ext set CWMin_           15
Mac/802_11Ext set CWMax_           1023
Mac/802_11Ext set SlotTime_        0.000009
Mac/802_11Ext set SIFS_            0.000016
Mac/802_11Ext set ShortRetryLimit_ 7
Mac/802_11Ext set LongRetryLimit_  4
Mac/802_11Ext set HeaderDuration_  0.000020
Mac/802_11Ext set SymbolDuration_  0.000004
Mac/802_11Ext set BasicModulationScheme_ 0
Mac/802_11Ext set use_802_11a_flag_ true
Mac/802_11Ext set RTSThreshold_    0
Mac/802_11Ext set MAC_DBG          0
```

```
Phy/WirelessPhyExt set HeaderDuration_        0.000020
Phy/WirelessPhyExt set Pt_                    $val(power)
Phy/WirelessPhyExt set freq_                  5.18e+9
Phy/WirelessPhyExt set BasicModulationScheme_ 0
Phy/WirelessPhyExt set trace_dist_            1e6
Phy/WirelessPhyExt set PHY_DBG_               0
Phy/WirelessPhyExt set PreambleCaptureSwitch_ 1
Phy/WirelessPhyExt set SINR_PreambleCapture_  2.5118
Phy/WirelessPhyExt set DataCaptureSwitch_     1
Phy/WirelessPhyExt set SINR_DataCapture_      100.0
Phy/WirelessPhyExt set L_                     1.0
Phy/WirelessPhyExt set CSThresh_            [expr ($val(power)/($val(
   CSrange)*$val(CSrange)*$val(CSrange)*$val(CSrange))) ];
Phy/WirelessPhyExt set PowerMonitorThresh_  [expr ($val(power)/($val(
   PMrange)*$val(PMrange)*$val(PMrange)*$val(PMrange))) ];
Phy/WirelessPhyExt set noise_floor_         $val(Noise)

#=============================
# Simulation Parameters Setup
#=============================
set val(chan) Channel/WirelessChannel
set val(prop) Propagation/TwoRayGround
set val(netif) Phy/WirelessPhyExt
set val(mac) Mac/802_11Ext
set val(ifq) Queue/DropTail/PriQueue ;# interface queue type
set val(ll) LL ;# link layer type
set val(ant) Antenna/OmniAntenna ;# antenna model
set val(ifqlen) 50 ; # max packet in ifq
set val(rp) DumbAgent;# routing protocol
set val(stop) 15.0 ;
set val(x) $val(Grid) ;
set val(y) $val(Grid) ;
set val(maxpkt) 10000 ;#[lindex $argv 3];
set val(pktsize) 1500 ;#[lindex $argv 4];
set val(rate) 6000kb ;#[lindex $argv 6];
set val(rseed) [clock seconds];
set filename Den-trace-$val(seq)-[lindex $argv 1]; #$val(nn)

#================
# Initialization
#================
# Create a ns simulator
set ns_ [new Simulator]
# Create a ns simulator
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
set god_ [create-god [expr $val(nn)]]
set tracefd [open $filename.tr w]
$ns_ trace-all $tracefd
$ns_ use-newtrace
set chan [new $val(chan)]

#=============================
# Mobile Node Parameter Setup
```

```
#=============================
$ns_ node-config -adhocRouting $val(rp) \
-llType $val(ll) \
-macType $val(mac) \
-ifqType $val(ifq) \
-ifqLen $val(ifqlen) \
-antType $val(ant) \
-propType $val(prop) \
-phyType $val(netif) \
-channel $chan \
-topoInstance $topo \
-agentTrace OFF \
-routerTrace OFF \
-macTrace OFF \
-movementTrace OFF \
-phyTrace OFF

#==================
# Nodes Definition
#==================
# Create nodes
for {set i 0} {$i < $val(nn) } {incr i} {

  if { $i == 0 } {
    $ns_ node-config  -agentTrace OFF -routerTrace OFF -macTrace ON -
       movementTrace OFF -phyTrace ON
  } else {
      $ns_ node-config  -agentTrace OFF -routerTrace OFF -macTrace OFF
         -movementTrace OFF -phyTrace OFF
  }
  set ID_($i) $i
  set node_($i) [$ns_ node]
  $node_($i) set id_  $ID_($i)
  $node_($i) set address_ $ID_($i)
  $node_($i) nodeid $ID_($i)
}

set val(sc) "./Den-scenario$val(Scen)"
source $val(sc)
set val(cb) "./Den-cbrtraffic$val(Scen)"
source $val(cb)

#========================
# Simulation Termination
#========================
proc finish { nf seq } {
global ns_ tracefd filename
$ns_ flush-trace
close $tracefd
exec awk -v nf=$nf -v SEQ=$seq -f Trace.awk $filename.tr &
exit 0
}
for {set i 0} {$i < $val(nn)} {incr i} {
$ns_ at $val(stop) "\$node_($i) reset"
}
$ns_ at $val(stop) "finish $val(nf) $val(seq)"
$ns_ at $val(stop) "puts \"done\" ; $ns_ halt"
$ns_ run
```

B.2 Scenario.tcl

This file illustrates Node grid.

```
# Scenario.tcl
# Program to make nodes randomly distributed in Grid
# By Byungjin Cho


#=================
# Set random seed
#=================
set opt(seed) [clock seconds] ;#clicks -milliseconds];

#==================================
# Procedure for optional parameters
#==================================
proc usage {} {
     global argv0
     puts "\nusage: $argv0 \[-nn nodes\] \[-seed seed\] \[-grid grid\]\
        n"
}

#=====================================
# Procedure to get optional parameters
#=====================================
proc getopt {argc argv} {
     global opt
     lappend optlist nn seed grid
     for {set i 0} {$i < $argc} {incr i} {
          set arg [lindex $argv $i]
          if {[string range $arg 0 0] != "-"} continue
                 set name [string range $arg 1 end]
                 set opt($name) [lindex $argv [expr $i+1]]
      }
}

#======================================
# Procedure to generate random location
#======================================
proc create-cbr-connection { a b } {
     global rng opt
     # bcho modified 21/11/2010
     set angle  [$rng uniform 0.0 1.0]
     set radius [$rng uniform 0.0 1.0]
     set randX  [expr ($b/2)+ $b/2*sqrt($radius)*cos($angle*2*3.14159)]
     set randY  [expr ($b/2)+ $b/2*sqrt($radius)*sin($angle*2*3.14159)]
     puts "\$node_($a) set X_  $randX"
     puts "\$node_($a) set Y_  $randY"
}

#======================
# Set random parameters
```

```
#=======================
getopt $argc $argv
set rng [new RNG]
$rng seed $opt(seed)

#===============
# Main Program
#===============
for {set i 0} {$i < [expr $opt(nn) ] } {incr i} {
    if {$i == 0 } {
        puts "\$node_(0) set X_  [expr $opt(grid)/2]"
        puts "\$node_(0) set Y_  [expr $opt(grid)/2]"
    } else {
        create-cbr-connection $i $opt(grid)
    }
}
```

B.3 Traffic.tcl

This file illustrates traffic generation.

```
# Traffic.tcl
# Programe to generate random traffic
# Modified by Byungjin Cho from CMU's Mobile code

# =================================
# Procedure for optional parameters
# =================================
proc usage {} {
    global argv0

    puts "\nusage: $argv0 \[-type cbr|tcp\] \[-nn nodes\] \[-seed seed
       \] \[-mc connections\] \[-rate rate\]\n"
}

# ====================================
# Procedure to get optional parameters
# ====================================
proc getopt {argc argv} {
    global opt
    lappend optlist nn seed mc rate type

    for {set i 0} {$i < $argc} {incr i} {
        set arg [lindex $argv $i]
        if {[string range $arg 0 0] != "-"} continue

        set name [string range $arg 1 end]
        set opt($name) [lindex $argv [expr $i+1]]
    }
}
# #############
# CBR Procedure
# #############
proc create-cbr-connection { src dst } {
    global rng cbr_cnt opt

    set stime [$rng uniform 0.0 0.35] ;# 180.0
    puts "#\n# $src connecting to $dst at time $stime\n#"
        if {$src == 0 } {
            set src [expr $opt(nn)-2]
        }
        if {$dst == 0 } {
            set dst [expr $opt(nn)-2]
        }
        puts "set udp_($cbr_cnt) \[new Agent/UDP\]"
    puts "\$ns_ attach-agent \$node_($src) \$udp_($cbr_cnt)"
    puts "set null_($cbr_cnt) \[new Agent/Null\]"
    puts "\$ns_ attach-agent \$node_($dst) \$null_($cbr_cnt)"
    puts "set cbr_($cbr_cnt) \[new Application/Traffic/CBR\]"
    puts "\$cbr_($cbr_cnt) set packetSize_ $opt(pktsize)"
    puts "\$cbr_($cbr_cnt) set rate_ $opt(rate)"
    puts "\$cbr_($cbr_cnt) set random_ 1"
    puts "\$cbr_($cbr_cnt) set maxpkts_ $opt(maxpkt)"
    puts "\$cbr_($cbr_cnt) attach-agent \$udp_($cbr_cnt)"
    puts "\$ns_ at $stime \"\$cbr_($cbr_cnt) start\""
    puts "\$udp_($cbr_cnt) set packetSize_ 3000"
    puts "\$ns_ connect \$udp_($cbr_cnt) \$null_($cbr_cnt)"
    incr cbr_cnt
}

# #############
# TCP Procedure
# #############
proc create-tcp-connection { src dst } {
    global rng cbr_cnt opt
    set stime [$rng uniform 0.0 10.0]
    puts "#\n# $src connecting to $dst at time $stime\n#"
    puts "set tcp_($cbr_cnt) \[\$ns_ create-connection \
        TCP \$node_($src) TCPSink \$node_($dst) 0\]";
    puts "\$tcp_($cbr_cnt) set window_ 32"
    puts "\$tcp_($cbr_cnt) set packetSize_ $opt(pktsize)"
    puts "set ftp_($cbr_cnt) \[\$tcp_($cbr_cnt) attach-source FTP\]"
    puts "\$ns_ at $stime \"\$ftp_($cbr_cnt) start\""
    incr cbr_cnt
}

# ############
# Main Program
# ############
getopt $argc $argv
if { $opt(type) == "" } {
    usage
    exit
} elseif { $opt(type) == "cbr" } {
    if { $opt(nn) == 0 || $opt(seed) == 0.0 || $opt(mc) == 0 || $opt(
       rate) == 0 } {
    usage
    exit
    }
    if { $opt(interval) ≤ 0.0 } {
    puts "\ninvalid sending rate $opt(rate)\n"
    exit
    }
}
puts "#\n# nodes: $opt(nn), max conn: $opt(mc), send rate: $opt(
   interval), seed: $opt(seed)\n#"
set rng [new RNG]
$rng seed $opt(seed)
set u [new RandomVariable/Uniform]
$u set min_ 0
$u set max_ 100
$u use-rng $rng
set cbr_cnt 1
set src_cnt 0
```

```
for {set i 1} {$i < [expr $opt(nn) ] } {incr i} {
    set x [$u value]
    if {$x < 50} {continue;}
    incr src_cnt
    set dst [expr ($i+1) % [expr $opt(nn) ] ]
    if { $opt(type) == "cbr" } {
        create-cbr-connection $i $dst
    } else {
        create-tcp-connection $i $dst
    }
    if { $cbr_cnt == $opt(mc) } {
        break
    }
    if {$x < 75} {continue;}
    set dst [expr ($i+2) % [expr $opt(nn) ] ]
    if { $opt(type) == "cbr" } {
        create-cbr-connection $i $dst
    } else {
        create-tcp-connection $i $dst
    }
    if { $cbr_cnt == $opt(mc) } {
        break
    }
}
```

B.4 Trace.awk

This file illustrates how interference information is extracted from Trace file.

```
# Trace.awk
# Extract interference power information in Trace file
# Modified by Byungjin Cho


#! /usr/bin/awk -f
#
# Parse a ns2 wireless trace file and generate the following stats:
# - number of flows (senders)
# - time of simulation run
# - number of packets sent (at the Application)
# - number of packets received (at the Application)
# - number of packets dropped (at the Application)
# - number of collisions (802.11)
# - average delay
# - average throughput
# - average traffic rate (measured)
#
# Author: ?   - I don't remember where I found the first version of
   this script :-(
# Modified by Julian Monteiro <jm@ime.usp.br>
#
#-------------------------------------------------------
function average (array) {
        sum = 0;
        items = 0;
        for (i in array) {
        sum += array[i];
        items++;
        }
        if (sum == 0 || items == 0)
        return 0;
        else
        return sum / items;
        }

function average2 (array, substract) {
    sum = 0;
    items = 0;
    for (i in array) {
        sum += array[i];
        items++;
    }
    if (sum == 0 || items == 0)
        return 0;
    else
        return sum / (items-substract);
}
#-------------------------------------------------------
```

```
function max( array ) {
    begin=1;
    for (i in array) {
        if (begin || array[i] > largest){
            largest = array[i];
            begin = 0;
        }
    }
    return largest;
}
#------------------------------------------------------
function min(array) {
    begin=1;
    for (i in array){
        if (begin || array[i] < smallest) {
            smallest = array[i];
            begin = 0;
        }
    }
    return smallest;
}
#------------------------------------------------------
function std_deviation(array, avg) {
    total = 0;
    items = 0;
    for (i in array) {
        △ = array[i] - avg;
        total += △*△;
        items++;
    }
    if (total == 0 || items == 0)
        return 0;
    else
        return sqrt(total/(items-1));
}
##=====================================================
BEGIN {
    if (!NEWTRACE) NEWTRACE="true";
    if (!PKTSIZE) PKTSIZE = 1500;
    if (!NODE_INITIAL_ENERGY) NODE_INITIAL_ENERGY = 100.0;
    sinr_count=0;
    total_packets_sent = 0;
    total_packets_received = 0;
    total_packets_dropped = 0;
    first_packet_sent = 0;
    last_packet_sent = 0;
    last_packet_received = 0;
    CTS_drop = 0;
    count=0;
}
# ======================================================
{
    if (NEWTRACE == "true") {
     event = $1;
```

```
        time = $3;
        node = $5;
        nodesrc = $7;
        nodeid = $9;
        type = $19;
        reason = $21;
        packetid = $43;
        packettype = $37;
        src = $33;
        dst = $35;
        cbr_packetid = $49;
        numhops = $51;
        opt_numhops = $53;
        energy  = $17;
        mac_type = $29;
        mac_src = $27;
        mac_dst = $25;
        rtsid = $31;
# strip  trailing .0 or :0 from src and dst
        sub(/\.0$/, "", src);
        sub(/\.1$/, "", src);
        sub(/\.2$/, "", src);
        sub(/\.3$/, "", src);
        sub(/\.0$/, "", dst);
        sub(/\.1$/, "", dst);
        sub(/\.2$/, "", dst);
        sub(/\.3$/, "", dst);

    } else {
       event = $1;
        time = $2;
        node = $3;
        type = $4;
        reason = $5;
        packetid = $6;
        packettype = $7;
        src = $14;
        dst = $15;
        cbr_packetid = $6; #ESTA ERRADO! :-(
        numhops = $19;
        opt_numhops = $20;
        sub(/^_*/, "", node);
        sub(/_*$/, "", node);
        sub(/^\[/, "", src);
        sub(/\:0$/, "", src);
        sub(/\:0$/, "", dst);
     }
    if ( time < simulation_start || simulation_start == 0 )
                simulation_start = time;
    if ( time > simulation_end )
                simulation_end = time;
#-------------
#--- DROP
#------------
```

```
    if (  ((time ≥ 9)&&(time < 14.0))  && (nodeid == 0 )
                                  && ( event == "r" || event == "d" )
                                     ) {
        count++;
        srcdst = mac_src"-"mac_dst;
        sinr_count = srcdst" "rtsid""count;
        rts_sinr[sinr_count]  =  energy;
    }
}# begin

END {

    number_flows = 0;
    sum_pkt = 0;
    avg_collision = 0;
    sum_rts = 0;
    sum_rts_flow = 0;
    avg_rts_collision = 0;
    substract=0;
    sum_sinr = 0;
    count_sinr = 0;
    file_sinr = sprintf("%s-SINR.dat", SEQ);
    for (count in rts_sinr) {
        rts_interf[count] = rts_sinr[count]-nf;
        if ((rts_interf[count] > 0.0000) && (count_sinr < 10000)) {
          printf("%f\n",rts_interf[count])>>file_sinr;
          sum_sinr = sum_sinr + rts_interf[count];
          count_sinr++;
        }

    }
}
```

B.5 Poisson.c

This file illustrates Poisson distribution. It is compiled by the following command [gcc -g -o Poisson -lm Poisson.c].

```
// Poisson.c
// Program for Poisson distribution
// By Byungjin Cho

#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <time.h>
#include <sys/time.h>

int lambda = 0;
static double _fact(int n);
static double _pmf(int n);
static double _cdf(int n);
static int _icdf(double p);

int main(int argc, char *argv[])
{
    double p = 0;
    int n;
    struct timeval tv;
    if (argc < 2)
    {
        fprintf (stderr, "invalid parameter\n");
        return -1;
    }
    lambda = atoi(argv[1]);
    gettimeofday(&tv, NULL);
    srand((unsigned int)tv.tv_usec * tv.tv_sec);
    p = ((double)rand()/RAND_MAX);
    n = _icdf(p);
    fprintf(stdout, "%d", n);
    return n;
}

static double _pmf(int n)
{
    double ret = 0.0;
    if (n < 0)
    {
        ret = 0.0;
    }
    else if (n == 0)
    {
        ret = exp((double)lambda * (-1));
    }
    else
    {
        double ret1 = (double)n * log((double)lambda);
        double ret2 = -1 * (double)lambda;
        double ret3 = 0;

        int k = 0;

        for (k = 1; k ≤ n; k++)
            ret3 = ret3 - log((double)k);

        ret = exp(ret1 + ret2 + ret3);
    }

    return ret;
}

static double _cdf(int n)
{
    double ret = 0.0;
    int x;

    if (n < 0)
    {
        ret = 0.0;
    }
    else
    {
        for (x = 0; x ≤ n; x++)
        {
            ret += _pmf(x);
        }
    }

    return ret;
}

static int _icdf(double p)
{
    int x;
    double p0;

    if (p < 0.0 || p > 1.0 || isnan(p))
    {
        return INT_MIN;
    }

    if (p == 0.0)
    {
        return -1;
    }

    if (p == 1.0)
    {
        return INT_MAX;
    }

    x = lambda;
    p0 = _cdf(x);
    while(p0 ≤ p && x < INT_MAX)
    {
        p0 = _cdf(++x);
    }
    while (p0 > p && x ≥ 0)
    {
        p0 = _cdf(--x);
    }

    return x + 1;
}
```