EDGF: Empirical dataset generation framework for wireless network networks

Dinesh Kumar Sah, Praveen Kumar Donta Tarachand Amgoth

Department of Computer Science and Engineering
Indian Institute of Technology (Indian School of Mines), Dhanbad
Jharkhand, India-826004
dksah.iitd@gmail.com, praveenisn@gmail.com tarachand.ism@gmail.com

Abstract

In wireless sensor networks (WSNs), simulation practices, system models, algorithms, and protocols have been published worldwide based on the assumption of randomness. The applied statistics used for randomness in WSNs are broad in nature, e.g., random deployment, activity tracking, packet generation, etc. Even though with adequate formal and informal information provided and pledge by authors, validation of the proposal became a challenging issue. The minuscule information alteration in implementation and validation can reflect the enormous effect on eventual results. In this proposal, we show how the results are affected by the generalized assumption made on randomness. In sensor node deployment, ambiguity arises due to node error-value (ε), and it’s upper bound in the relative position is estimated to understand the delicacy of diminutives changes. Moreover, the effect of uniformity in the traffic and contribution of scheduling position of nodes also generalized. We propose an algorithm to generate the unified dataset for the general and some specific applications system models in WSNs. The results produced by our algorithm reflects the pseudo-randomness and can efficiently regenerate through seed value for validation.

Keywords: Dataset generation, random deployment, wireless network, clustering, traffic data.

1. Introduction

In wireless sensor networks (WSN) research, the correctness of simulation practices and results are important because of the limitation occurs implementations and testbeds. There are many pseudo-random numbers generators (PNRG) are available such as simulators Mersenne Twister [1], Xorshift [2, 3], linear congruential generators [4, 5, 6, 7, 8] and so forth [9, 10]. The PNRG produces 32-bit or 64-bit words, which can further decode as integer based on the requirement. In most of the simulation or programming tools, “Mersenne Twister” [1] has been used to generate the random numbers (RN). The problem that arises in PRNG is that even with the same seed value, the reproduction of the same sequence is almost impossible.
Several existing proposals in WSNs result, use random deployment to validate connectivity, packet generation, and much more use this function regularly. These practices raise a concern about the validation of results. Often formal description seems missing in the proposal, and provided function not enough to reproduce. Moreover, the exploration of random processes provided by simulator and programming tools is also tedious, creating hurdles for authors.

This work proposes a new dataset generation framework for WSNs referred to as empirical data generation framework (EDGF). The objective is to provide the unified datasets which are empirical for validation purpose. Our function’s main contribution is that it uses a modified version of the linear congruential function with internal seed value based on the user input. We also observe some useful instances through experiments: the seed constants, provide string deployment automatically and report in the result analysis. Moreover, our proposal is independent of the limitation imposed on the congruential generator of selection of prime \( m \) and selection of \( a \) as a primitive element modulo \( m \).

1.1. Contributions

The contributions of the work are summarized as follows.

1. We propose a new deterministic algorithm to generate datasets for sensor node deployment and packet generation. The objective is to provide the dataset for 2-D deployment coordinates, and the traffic matrix corresponded to the deployed nodes.

2. Deployment data generation with the critical aspects of function and seed value.

3. Traffic generation data for uniform and exponential traffic matrix. The significance of exponential traffic is that it is needed for event-driven applications.

4. Randomness property validation of EDGF data with KS-test, \( \chi^2 \)-test and auto-correlation test to assure the uniformity and randomness.

5. Illustration of datasets produced by EDGF and highlighting important issues.

2. Effect of random datasets in WSNs

In this section, we will cover the significance of the randomness and its effect on the validation. Here, randomness on deployment and data traffic are considered because these are standard requirements of almost all the WSNs models. However, we are in the initial stages of EDGF; therefore, we are discussing datasets for deployment and packet generation only in this work. We also observe that our framework’s true potential is limitless, and many other datasets can also be generated using the EDGF for various applications in WSNs.

Most of the time, linear congruential generator and middle square method used to generate RN for deployment. In computing and programming, the algorithm can produce an extensive RN sequence and
can determine by the initial input is called seed-based PRNG. By knowing the seed value, the entire RN sequence can regenerate, which is often a requirement of computer protocols. This type of RN generator is often called a pseudo-random number generator (PRNG). Moreover, the RN generated by the PRNG takes the system clock as seeds value. In our example, we are taking 2-D random values on the X-Y plane generated by Python’s function (NetworkX-package) to depict the points as node’ location in deployment. To proceed with the effect, we perform a K-mean clustering algorithm to form a cluster on those points. In many of the WSNs application practices, the clustering operation is often used due to its usefulness in energy, lifetime, congestion, and many more. We further estimate the euclidean distance among the cluster head and sink. After estimation, it is easy to observe the values often show the significant difference among them. Note that the distance among nodes is directly proportional to the energy consumption; transmission range estimation affects the network performance.

To analyze the effect of randomness in data traffic generation, we consider CSMA and TDMA-MAC to explore the effects. In [12], CSMA/∗p has been considered based on the optimal probability distribution of the traffic produced by the nodes. The constraint is that the number of nodes and channel access rate of nodes is supposedly known for this protocol to enhance channel utilization enhancement. Moreover, an extension of [12] another proposal name “sift” [13] been given based on CSMA/p∗ which also provide better utilization in the case of availability of data at the nodes. If the data arrival of nodes is random, it is almost impossible to sense the transmission of more than two hops. The performance also starts degrading only because of the randomness.

In Z-MAC [14], distributed algorithm DRAND [15] extension of centralized protocol RAND [16] used to assign the time slots to each node for synchronization to perform the node schedule further. As synchronization is one of the important aspects of node scheduling, it’s highly anticipated that most of the
synchronization algorithms consider the hop localization to assign the slot number. If the node position at the time of deployment is taken randomly, then the slots will vary throughout the validation. Further, to enhance the usability of unused slots, an interesting mechanism has been given. Let suppose, we have 4 node a,b,...,j, need to schedule in order of a,b,c,...,j. Now all the node is the owner of its time slot and free to transmit its packet through TDMA. If any owner node is not using its slot, then those slots will be eligible for CSMA data transmission. The effect of uniform random packet generation is that suppose if any 2nd slot and 9th slot is not used by its owner. Then in these instance, the congestion on 2nd will be \( \sum_{\text{leftover packets}}(2 \rightarrow \text{onwards}) \) and it must be higher than \( \sum_{\text{leftover packets}}(9 \rightarrow \text{onwards}) \) in uniform traffic scenario. These instances can have a ripple effect in both ways, but one thing for sure that the packet transmission rate will be different in both instances. Now, at the time of validation due to congestion, the same algorithm’s packet delivery rate will be different for the same random function. This section has shown the observable evidence of how the random function can cause a severe difference in result validation, even though using the same procedure for deployment and packet generation.

3. Empirical dataset generation framework

In this section, we develop EDGF for WSNs scenario for node deployment and packet generation. The significance of randomness in the dataset such as periods, uniformity, and independence, are covered carefully for both instances. In deployment, apart from randomness, the other issues such as localization, connectivity, outliers handling mechanism, and coverage are considered to make it as generalized as possible. To determine the packet generation of the individual nodes, the proportionality of the packet generation needs to identify first. In \cite{17}, the interval of the packet generation rate and its distribution throughout the networks has been shown.

3.1. Topological significance of datasets

In WSNs, the deployment of nodes represented as graph \( G \) with nodes, \( n_1, n_2, \ldots, n_i \) and the coordinate value in 2-D plane represented as \( n_{1,x,y}, n_{2,x,y}, \ldots, n_{i,x,y} \). The topological significance of the network follows the same properties as the graph. The graph is said to be connected if there exists a path between every pair of nodes. For a connected graph with \( i \) number of nodes, \( i(i-1)/2 \) edges are needed to make the path between every pair of nodes. In WSNs, very little certainty that each node falls into the range of every other node. Though the network connectivity is a different problem, many random generated functions do not care to provide the points that can form the connected graph within the sensing range \( R \). One simple solution is to increase the sensing range of each node, but energy consumption will vary at the validation time. The EDGF identified these issues and pointed out the number of isolated nodes with individual sensing range values. However, EDGF is able to provide the coordinate based on a non-hierarchical manner suitable for mesh topology.
We observe some critical issues like connectivity, coverage, and localization are not full-filled by the value generated through the random function because of the user’s lesser control. Therefore, we are merely highlighting the issues and leaving as open problems. The nodes’ output and position are provided, which can further incorporate as input to handle connectivity, coverage maximization, and localization in networks. In [6], many random generators have been explored, out of which we have opted \( x_n := (a(x_n - 1) + c) \mod m \), though changing the constant does not going to change the vector. With this selection, only two possibilities left as either change the constant with some good suggestion given in [11] or combined it with some other generator integrated multiple-recursive generator (CMRG) as in [18]. Moreover, another combination is the \( a \) and \( c \) value can be a real number. The possibilities \( a \) and \( c \) are a real number is limitless, and floating operation is also expensive. Moreover, with some limited number of test instance and universal mathematical constant values [19], the results are impressive. Though the vast exploration and validation require to check the limit of the function, it seems to pass in the initial tests performed for deployment. Our function’s advantage is that it has less memory requirement as it does not need to store the constant value in term of \( 2^{32} \) or more to maintain the period of randomness. Moreover, our deployment’s complexity is also \( O(n) \), which is enough to proceed the path forward.

In algorithm 1 to 4, the input set is defined as \( n \) number of sensor nodes, \( m \) area of deployment in unit\(^2 \) as \( m \times n \) where \( m \) and \( n \) can be width and length, \( X[0] \) is seed value for \( X \) and \( Y \) -coordinate, \( mcV \) is the list of mathematical constant values defined internally based on the seed value given as input. Moreover, \( a \) and \( c \) are constant generated using the function \( a = mcV[X[0] \% |mcV|] \), \( c = mcV \left( (X[0] + \frac{|mcV|}{2}) \% |mcV| \right) \) and \( a \neq c \). Moreover, for packet generation, the number of time slots needed and defined as \( t \), with the number of packets generation range, vary between the \( P_1 \) and \( P_2 \).

3.2. Grid deployment

Here, the objective is to generate the empirical dataset for two-dimensional space, such that it obey the simple rules of randomness and networks. The first requirement is it should be surely random within the range of the area of interest. The WSNs properties such as connectivity and coverage of the network should be maintained. In some instances, outliers exist in the network, so there must be the proper outliers handling mechanisms. Moreover, to deals with the outliers, the relaxation in the term of error-value (\( \epsilon - value \)) is being made to accommodate the outliers nodes in the network to maintain the connectivity. Meanwhile, the following relaxation is being optional in the algorithm with the concern that some instance such as mobile sink able to handle the outliers and hidden nodes. Moreover, through the experimental observations, the uniformity of node deployment is enhanced with the adoption of grid deployment. In our work, grid is defined as the area such as if we plot the axis by taking the coordinate \((m/2, n/2)\). The entire grid area \( G \) further divided into four sub-grids \( G = g_1, g_2, g_3, g_4 \). The advantage of this assumption that it increase the connectivity and coverage of network without any separate algorithm.
Algorithm 1: Deployment data generation without grid Algorithm

Data: (n, m, X[0])

Result: Deployment coordinates (X,Y) in range R

1 begin
2 // a and c are constant a ≠ c
3 4
4 a = mcV[X[0]%|mcV|]
5 c = mcV [(X[0] + mcV)|mcV|] %|mcV|
6 Y[0]=X[0] // y coordinate seed value
7
8 // Deployment of Sensors
9 for i=1 to n do
10 X[i]=(a*X[i-1]+c)%m
11 Y[i]=(a*Y[i-1]+a)%m
12
13 // Make Graph
14 for i=1 to n do
15 G.add_node(Point(X[i],Y[i]))

4. Packet generation

There are two class of data or packet traffic for WSN applications, continuous monitoring or event driven
monitoring. The requirement of packet traffic may vary along with the applications.

4.1. Uniform Packet generation

In continuous monitoring applications, the sensors send the packets in every time interval. These activity
can be model as uniform distribution of packets throughout the network within the specific range.

4.1.1. lemma

Consider a continuous monitoring system, the packet rate of sensors model as uniform distribution traffic.

4.1.2. proof

If \( x_i \) is a value taken from the TUD, then the value \( a + (b - a)x_i \) follows the TUD constrain by \( a \) and \( b \). The probability, that a TUD random variable can categorized in the interval of with the fixed range of
variables is independent of the position in the interval itself (At the same it depend upon the interval size).
To observe this, if $X$ is uniformly distributed over $U(a, b)$ and $[x, x + d]$ is a subinterval of $[a, b]$ with fixed $d > 0$, then $P(X \in [x, x + d]) = \int_x^{x+d} \frac{dy}{b-a} = \frac{d}{b-a}$ which is independent of $x$.

4.2. Exponential data generation for event driven monitoring (EDM)

In EDM, the sensors are being activate only when some activity occurs in sensing range. These are the scenario when sensors can exploit the correlation to improve the network efficiency by adopting the data acquisition mechanism. In [20], the network management has been suggested based on the spatial-temporal (ST) relation of sensors. The advantage of adoption of ST is that it can ensure the effect of any event driven activity and can distribute that among the nodes exponentially. Let assume that at any time instance $t$ some event $e_t$ occurs in grid $g_i$ and in locality $l_2, g_i$ which effect is monitored by sensor in those locality. Now, as the point of activity and individual sensor $s_i$ distance being increase, the effect start fading up. There is the possibilities that the corner most sensors of other grid in different locality such as $l_1, g_{i+1} \% 4, l_2, g_{i+2} \% 4, l_3, g_{i+3} \% 4, l_4, g_{i+3} \% 4$ will not even able to sense the event due to the limitation on the sensing range. In this instance the uniform generation of data packets in simulation practices might lead to ambiguous result. Therefore, we are giving the algorithm to generate the data packet with exponential distribution. Let assume the $F(x)$ be the exponential function distribute over random variable $x$ and rate $\lambda$ and define as:

$$F(x) = \begin{cases} 1 - e^{-\lambda x}, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

(1)

Here $\lambda > 0$ is the rate parameter of the exponential. Now we already generated $r_{1,t}, r_{2,t}, \ldots, r_{n,t}$ through algorithm 3 which individual input value $r_i = 1 - e^{-\lambda x_i}$ for $\lambda = 1$. Further, $-\lambda x_i = ln(1 - r_i)$ which will generate the individual $x_i = -\frac{1}{\lambda}ln(1 - r_i)$. (Generally $x_i$ w.r.t $r_i$). Now, the minimum effect or null effect for the nodes which are further away from the event can also prove to be random (see Appendices A.1).

4.2.1. lemma

Consider a correlation aware monitoring system, the packet rate of sensors model to the considerably far nodes is also exponential distribution traffic.

4.2.2. proof

In Appendices A.1, the proof of minimum $\lambda$ is given which is similar to the nodes which is far from the activity point. The $\lambda$ will be small but still the data generated w.r.t the point shows exponential distribution.

5. Analysis and results

To establish the degree of agreement among the distribution of a sample of generated RN and theoretical uniform distribution (TUD), we are performing two well known for uniformity test and one for independence
1. Kolmogorov-Smirnov test (KS-test) (uniformity test)

2. $\chi^2$ test ($\chi^2$ Test) (uniformity test)

3. Autocorrelation test (independence test)

To analyze the RN generated for the test, two hypotheses made as one support the RN generator is indeed uniformly distributed. Then $H_0$, can define as a null hypothesis. The other one can support the RN generator is not uniformly distributed can represent as $H_1$, and act as alternative hypothesis. Depending upon the null hypothesis of no major difference among sample distribution and TUD we will conclude that whether the number generated through function is truly random with significance levels ($\alpha$).

In the experimental results, with respect to different constant $a, c$ and $KS - test(\alpha = 0.01)$, $\chi^2 - test(\alpha = 0.001)$, autocorelation($\alpha = 0.01$), the test results are tabulated in Appendices A.3, Table A.1.

In analysis, the hypothesis based test predict that whether the two mutually exclusive statements regarding the sample to figure out which statement is more supportive to sample data drawn from the population. Further, the test result is significance only if the sample data drawn from the population is large enough or frequent enough comparative to the $H_0$ to reject the $H_0$ for overall population. There is some instance of special concern in hypothesis test such as the assumption of $H_0 = true$, limit of $\alpha$-value, nature of sample data (large value drawn from critical region). We have to understand here that, test measurement based on $\alpha$ dose not support the acceptance as 100% accuracy but to not reject the nature (randomness in our case) with individual $\alpha$. The cases where either the $\alpha$ value is change or sample change to make sure the confidence of data but it will never be 100% accurate.

Most popular $\alpha$ values are lies in between $[0.01, 0.05]$ and taken in the account for our experiment too except one exception of $\alpha = 0.001$. Let suppose if $\alpha = 0.05$, expect to obtain sample means in the critical region 5% of the time when the $H_0$ is true. Then, we can not determine that the null hypothesis is true, but if it falls in the critical region then it get reject. This is the reason, $\alpha$ value consider sometime as error rate. In our test result if by providing the $\epsilon$-value relaxation in $\alpha$, if the test is pass than result can accepted. The true nature of $\epsilon$-value relaxation is also need to explore.

5.1. Kolmogorov-Smirnov test (KS-test) of generated datasets

Let suppose the sample of RN generator from our algorithm is $r_1, r_2, \ldots, r_n$ where the $n$ represent the total number of sample of RN generated. In KS-test, the sub-sample is being selected from the sample size in such a way that size of sub-sample $\leq n$. The hypothesis suggests that if the subsample passes the test for some individual $\alpha$ value, it means the number passes the uniformity test, and the test can further also
proceed for different sub-sample. The empirical \( s_n(x) \) is depended upon the sample size taken for test and can define:

\[
s_n(x) = \left( \frac{|r_{i}, r_{i+1}, \ldots, r_{i+j}| \leq x}{N} \right) \text{where, } N = \frac{n}{4} \text{ and } N \subset n
\]  

(2)

The KS-test define with the greatest absolute expectation among between \( f(x) \) and \( s_n(x) \), over the range of random variables and difference in parametric test (DPT) can define as

\[
DPT = |f(x) - s_n(x)|
\]

(3)

(See Appendices A.2 for detail)

5.2. \( \chi^2 \) test (See Appendices A.2 for detail)

Chi-Square goodness of fit test determines if a sample data have similarity with the population. Once the \( \chi^2 \) value is calculated, then with respect to the degree of freedom and \( \alpha \) value, the individual \( \chi^2 \) value is being check and if the value is larger then the hypothesis is not reject.

5.3. Auto correlation test (See Appendices A.2 for detail)

It concerned with the dependencies between numbers in sequence to check whether it is any dependency on the number or not. Once the \( Z_0 \) value computed, it will compare with the Cumulative standardized normal distribution table and if \( Z_0 \leq Table(Z_0) \) w.r.t \( \alpha \), then the conclusion can be made that there is no alpha and hypothesis should be accepted. Apart from the auto-correlation, if the \( x \) and \( y \) value is in same dimension (which is true in case of deployment), circular co-relation has been suggested for a periodic sequence in [21].

Test result of the sample data-set of experiment is available in Appendices A.3 Table. A.1.

5.4. Results

Fig. 2 represent both grid and non-grid based random deployment of sensor nodes in a 100 sq. m area with the seed value 43 and different transmission ranges. In Grid-based deployment, we generate random position in one quarter and repeat the same positions in remaining quarters. Grid-based deployment makes easier to test the data set, and satisfies the connectivity of the nodes and covering the area. Fig. 2a and 2b shows the grid-based deployment with transmission range 10 and 15 respectively. The observation here we found that the sensors are well connected when increases the transmission range. The non-grid based random sensor deployment shown in Fig. 3a and 3b with transmission ranges 10 and 15. In non-grid based random deployment we found that the sensor placements are varied and the probability of placing more number of nodes in particular portion is high. The constant values such as \( a \) and \( c \) will get more impact to get the random deployment. We need to make some difference between these two constant values. If
Figure 2: Deployment of the network in grid and non-grid modes with $X[0]=43, a=3.359885666, b=1.902160583$ (a) Transmission range=10 (Grid-based) (b) Transmission range=15 (Grid-based) (c) Transmission range=10 (NonGrid-based) (d) Transmission range=15 (NonGrid-based) (e) $a=c$ (Grid-based) (f) $a=c$ (Nongrid-based)

we make both $a$ and $c$ are same then the result will be like Fig. 2e and Fig. 2f, where Fig. 2e represents Grid-based deployment and Fig. 2f represents non-grid based deployment.

In Fig. 3 K-S test statics of performed simulation have shown for EDGF and cumulative distribution function (CDF) for deployment and packet generation data. The $DPT$ shown in equation. 3 can be easily observe in Fig. 3 as in Fig. 3a and Fig. 3b the blue line shows the CDF based value of randomness and the orange line is able to replicate the uniformity in both grid and non-grid deployment. In Fig. 3d and Fig. 3c the depiction of packet generation for different time slots $t_1, t_2, ..., t_5$ has shown. The packet data seems not passing the uniformity in Fig. 3d for some interval which shows the biasness of the of the function. Even though, for uniform distribution as in Fig. 3d reference CDF denoted with blue line have showing partial failure, the exponential packet generation value in Fig. 3c is showing promising difference.
Appendix A. Appendices

Appendix A.1. Nature of distribution of the $\lambda \to 0$ in exponential distribution

Let $X_1, X_2, \ldots, X_N$ is the independent exponentially distributed random variables with variable $\lambda_1, \ldots, \lambda_n$. Then $\min X_1, X_2, \ldots, X_n$ is also exponentially distributed, $\lambda = \lambda_1 + \ldots + \lambda_n$

$$P(\min X_1 \ldots X_n) = P(X_1 > x \ldots X_n > x) = \prod_{i=1}^{n} P(X_i > x)$$

(A.1)

$$P(\min X_1 \ldots X_n) = \prod_{i=1}^{n} \exp \left( -x \sum_{i=1}^{n} \lambda_i \right)$$

(A.2)

To ensure the distribution still obeying the exponential, the index of variable $k$ which nature $\lambda \to 0$ is
define as

\[ P(k | X_k = \min X_1..X_n) = \frac{\lambda_k}{\lambda_1 + \lambda_2 + ... + \lambda_n} \quad \text{(A.3)} \]

Appendix A.2. Validation Test Description

Let suppose \( X \) is uniformly distributed over the unit interval of \([0, 1]\) then cumulative distribution function (CDF) of \( X \) can define as:

\[
f(x) = \begin{cases} 
0 : & x < 0 \\
0 & 0 \leq x < 1 \\
1 : & x \geq 1 
\end{cases} \quad \text{(A.4)}
\]

Appendix A.2.1. Kolmogorov-Smirnov test:

Following steps require to follow:

step.1 Compute \( D^+ \)

\[
D^+ = \max_{1 \leq j \leq n} \left\{ \frac{i}{n} - r_i \right\} \quad \text{(A.5)}
\]

\[
D^- = \max_{1 \leq j \leq n} \left\{ r_i - \frac{i - 1}{n} \right\} \quad \text{(A.6)}
\]

step.2 Compute max among \( D^+ \) and \( D^- \) as

\[
D = [D^+, D^-] \quad \text{(A.7)}
\]

step.3 Locate in KS-table critical value of \( D_\alpha \) for specified \( \alpha \) in sample space.

Appendix A.2.2. \( \chi^2 \)-test

To perform the \( \chi^2 \) test, we need to perform following steps on the sample RV for validation.

step.1 Compute \( \chi^2 \)

\[
\chi^2 = \sum_{i=1}^{n} \left\{ \frac{(F_i - AE_i)^2}{AE_i} \right\} \quad \text{(A.8)}
\]

where \( F_i \) is frequency of number in the class \( i^{th} \) class from uniform distribution. \( AE_i \) is defined as absolute expected number in each class equal to \( N/n \) for equally space class.

step.2 Compute \( \nu \)

\[
\nu = \text{number of } i - 1 \quad \text{(A.9)}
\]

Sampling distribution of \( \chi^2 \) is approximately the \( \chi^2 \) distribution with \((n - 1)\) degree of freedom.
Appendix A.2.3. autocorrelation and circular autocorrelation -test

To compute the autocorrelation between every \(m\) number (\(m\) is known as lag), will start from the \(i^{th}\) number with \(m\) as interval and can define as \(i+m, i+2m, i+3m,...\). To proceeds with, the following step is being taken to compute the autocorrelation \(\rho_{im}\) between numbers \(R_i, R_{i+m}, R_{i+2m}, ..., R_{i+(M+1)}\) is to be found where \(M\) is largest integer as \(i+(M+1)m \leq N\).

An non-zero auto-correlation implies lack of dependence following detailed test appropriate.

\[ H : \rho_{im} = 0 \text{ no correlation} \]

\[ H_I : \rho_{im} \neq 0 \text{ correlation} \]

step.1 Compute \(Z_0\)

\[ Z_0 = \frac{\hat{\rho}_{im}}{\delta_{\hat{\rho}_{im}}} \quad (A.10) \]

step.2 Compute \(\hat{\rho}_{im}\)

\[ \hat{\rho}_{im} = \frac{1}{M+1} \sum_{k=0}^{M} [R_{i+KM}R_{i+(K+1)m}] - 0.25 \quad (A.11) \]

step.3

\[ \delta_{\hat{\rho}_{im}} = \sqrt{\frac{13M+7}{12(M+1)}} \quad (A.12) \]

If two sequence of RN is periodic with same period \(N\), then the circular correlation can define as

\[ \hat{\rho}_{im} = \frac{1}{N} \sum_{k=0}^{N-1} x(n)y(n-m) \quad (A.13) \]

\[ \delta_{\hat{\rho}_{im}} = \sqrt{\frac{13N+7}{12(N+1)}} \quad (A.14) \]

where \(m = 0, 1, ..., (N-1)\)

Note : Only \(m=0\) considered in experiment with \(\alpha = 0.001\) for circulation corelation

Appendix A.3. Test case results Table A.1

Appendix A.4. Packet generation data for exponential distribution Table A.2

References

[1] M. Matsumoto, T. Nishimura, Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator, ACM Transactions on Modeling and Computer Simulation (TOMACS) 8 (1) (1998) 3–30.

[2] A. B. Owen, Latin supercube sampling for very high-dimensional simulations, ACM Transactions on Modeling and Computer Simulation (TOMACS) 8 (1) (1998) 71–102.
[3] G. Marsaglia, et al., Xorshift rngs, Journal of Statistical Software 8 (14) (2003) 1–6.
[4] A. De Matteis, S. Pagnutti, Parallelization of random number generators and long-range correlations, Numerische Mathematik 53 (5) (1988) 595–608.
[5] G. Fishman, Monte Carlo: concepts, algorithms, and applications, Springer Science & Business Media, 2013.
[6] P. L’Ecuyer, Random numbers for simulation, Communications of the ACM 33 (10) (1990) 85–97.
[7] P. L’ecuyer, Tables of linear congruential generators of different sizes and good lattice structure, Mathematics of Computation of the American Mathematical Society 68 (225) (1999) 249–260.
[8] P. L’Ecuyer, F. Blouin, R. Couture, A search for good multiple recursive random number generators, ACM Transactions on Modeling and Computer Simulation (TOMACS) 3 (2) (1993) 87–98.
[9] P. L’Ecuyer, History of uniform random number generation, in: WSC 2017-Winter Simulation Conference, 2017.
[10] P. L’Ecuyer, Random number generation, in: Handbook of Computational Statistics, Springer, 2012, pp. 35–71.
[11] F. James, A review of pseudorandom number generators, Computer physics communications 60 (3) (1990) 329–344.
[12] Y. Tay, K. Jamieson, H. Balakrishnan, Collision-minimizing csma and its applications to wireless sensor networks, IEEE Journal on selected areas in Communications 22 (6) (2004) 1048–1057.
[13] K. Jamieson, H. Balakrishnan, Y. Tay, Sift: A mac protocol for event-driven wireless sensor networks, in: European workshop on wireless sensor networks, Springer, 2006, pp. 260–275.
[14] I. Rhee, A. Warrier, M. Aia, J. Min, M. L. Sichitiu, Z-mac: a hybrid mac for wireless sensor networks, IEEE/ACM Transactions on Networking (TON) 16 (3) (2008) 511–524.
[15] I. Rhee, A. Warrier, J. Min, L. Xu, Drand: Distributed randomized tdma scheduling for wireless ad hoc networks, IEEE Transactions on Mobile Computing 8 (10) (2009) 1384–1396.
[16] S. Ramanathan, A unified framework and algorithm for channel assignment in wireless networks, Wireless Networks 5 (2) (1999) 81–94.
[17] D. K. Sah, T. Amgoth, Parametric survey on cross-layer designs for wireless sensor networks, Computer Science Review 27 (2018) 112–134.
[18] P. L’ecuyer, R. Simard, E. J. Chen, W. D. Kelton, An object-oriented random-number package with many long streams and substreams, Operations research 50 (6) (2002) 1073–1075.
[19] S. R. Finch, Mathematical constants, Vol. 93, Cambridge university press, 2003.
[20] S. N. Das, S. Misra, Correlation-aware cross-layer design for network management of wireless sensor networks, IET Wireless Sensor Systems 5 (6) (2015) 263–270.
[21] S. Salivahanan, A. Vallavaraj, C. Gnanapriya, Digital signal processing, mcgraw-hill (2001).
Algorithm 2: Grid-based deployment data generation Algorithm

Data: (n, m, X[0])

Result: Deployment coordinates (X, Y) in an area m \times n

begin

1  \begin{align*}
   a &= mcV[X[0] \% mcV] \\
   c &= mcV \left(\frac{X[0] + \frac{mcV}{2}}{mcV}\right) \% mcV \\
   Y[0] &= X[0] \quad // y coordinate seed value
\end{align*}

2  m1 = \frac{m}{4} \quad //Here area dividing in to 4 grids.

3  n1 = \frac{n}{4}

4  // Deployment of Sensors in Grid-1

5  for i=1 to n1 do
6     X[i] = (a*X[i-1] + c) \% m1
7     Y[i] = (a*Y[i-1] + a) \% m1

8  // Deployment of Sensors in Grid-2

9  for j=1 to n1 do
10     X[i+j] = X[j] + m1
11     Y[i+j] = Y[j] + m1

12  // Deployment of Sensors in Grid-3

13  for k=1 to n1 do
14     X[i+j+k] = X[k] + m1
15     Y[i+j+k] = Y[k]

16  // Deployment of Sensors in Grid-4

17  for l=1 to n1 do
18     X[i+j+k+l] = X[l]
19     Y[i+j+k+l] = Y[l] + m1

20  // Make Graph

21  for i=1 to n do
22     G.add_node(Point(X[i], Y[i]))
Algorithm 3: Uniform packet generation \( r_{1,t}, r_{2,t}, \ldots, r_{n,t} \) for node 1, 2,.., \( n \) and for time \( t \)

**Data:** \((n, t, P_1, P_2)\)

**Result:** Uniform packet generation algorithm

1. \( \begin{align*}
\text{// } P_1 &= \text{Minimum size of packet} \\
\text{// } P_2 &= \text{Maximum Size of packet} \\
\text{// } a \text{ and } c &= \text{are constant } a \neq c \\
X[0] &= mcV[P_2 \%|mcV|] \\
a &= mcV[X[0] \%|mcV|] \\
c &= mcV \left[ (X[0] + \frac{|mcV|}{2}) \%|mcV| \right] \\
\text{for } i=1 \text{ to } n \text{ do} \\
\text{for } j=1 \text{ to } t \text{ do} \\
X[j \%t][0] &= a(a \ast X[j-1] + c)(P_2 - P_1) + P_1 \\
Y[i][j] &= [-1 \lambda \log(1 - \frac{X[j \%t][0]}{P_2}) \%(P_2 - P_1)] + P_1
\end{align*} \)

Algorithm 4: Exponential Packet Generation Algorithm

**Data:** \((n, t, P_1, P_2)\)

**Result:** Exponential packet generation (Sample result available in Appendices A.4, Table A.2)

1. \( \begin{align*}
\text{// } a \text{ and } c &= \text{are constant } a \neq c \\
X[0][0] &= mcV[P_2 \%|mcV|] \\
a &= mcV[P_1 \%|mcV|] \\
c &= mcV \left[ (X[0] + \frac{|mcV|}{2}) \%|mcV| \right] \\
\text{for } i=1 \text{ to } n \text{ do} \\
\text{for } j=1 \text{ to } t \text{ do} \\
X[i \%t][j \%t] &= (a \ast X[(i-1)\%t][(j-1)\%t] + c)(P_2 - P_1) + P_1
\end{align*} \)
## Table A.1: Test case results

| X[n] | a value | c value | TR=10 | TR=15 | TR=20 | KS-Test | Chi2Test | Auto correlation Test | TR=10 | TR=15 | TR=20 | KS-Test | Chi2Test | Auto correlation Test |
|------|---------|---------|-------|-------|-------|---------|---------|-----------------------|-------|-------|-------|---------|---------|-----------------------|
| 0    | 4.669202| 2.295587| 10    | 1     | 0     | Satisfied| Satisfied| Satisfied              | 15    | 0     | 0     | Satisfied| Satisfied| Satisfied              |
| 2    | 3.375823| 1.705211| 11    | 0     | 0     | Satisfied| Satisfied| Rejected              | 8     | 4     | 0     | Rejected| Satisfied| Satisfied              |
| 3    | 2.80777 | 1.324718| 8     | 2     | 0     | Rejected| Rejected| Satisfied              | 13    | 2     | 0     | Satisfied| Satisfied| Satisfied              |
| 5    | 2.584982| 1.41593 | 11    | 1     | 1     | Satisfied| Satisfied| Satisfied              | 12    | 0     | 0     | Satisfied| Satisfied| Satisfied              |
| 7    | 2.295587| 4.669202| 1     | 0     | 0     | Satisfied| Satisfied| Satisfied              | 11    | 0     | 0     | Satisfied| Satisfied| Satisfied              |
| 12   | 3.141593| 2.584982| 5     | 2     | 1     | Rejected| Satisfied| Satisfied              | 9     | 0     | 0     | Satisfied| Satisfied| Satisfied              |
| 14   | 4.669202| 2.295587| 8     | 2     | 0     | Rejected| Rejected| Satisfied              | 14    | 2     | 0     | Satisfied| Rejected| Satisfied              |
| 24   | 1.324718| 2.80777 | 11    | 2     | 2     | Satisfied| Satisfied| Satisfied              | 11    | 0     | 0     | Satisfied| Rejected| Satisfied              |
| 43   | 3.359886| 1.902161| 8     | 0     | 0     | Satisfied| Satisfied| Satisfied              | 17    | 1     | 1     | Satisfied| Satisfied| Satisfied              |
| 59   | 2.80777 | 1.324718| 8     | 1     | 1     | Satisfied| Satisfied| Satisfied              | 12    | 0     | 0     | Satisfied| Satisfied| Satisfied              |
| 65   | 1.705211| 3.275823| 6     | 0     | 0     | Satisfied| Rejected| Satisfied              | 8     | 0     | 0     | Satisfied| Rejected| Satisfied              |
| 70   | 4.669202| 2.295587| 11    | 3     | 0     | Satisfied| Satisfied| Satisfied              | 7     | 0     | 0     | Rejected| Satisfied| Satisfied              |
| 76   | 2.502908| 2.718282| 5     | 0     | 0     | Satisfied| Satisfied| Satisfied              | 9     | 3     | 0     | Rejected| Satisfied| Satisfied              |
| 87   | 2.80777 | 1.324718| 9     | 2     | 0     | Rejected| Satisfied| Satisfied              | 8     | 2     | 0     | Rejected| Satisfied| Satisfied              |
| 111  | 2.584982| 3.141593| 9     | 3     | 0     | Satisfied| Satisfied| Satisfied              | 8     | 0     | 0     | Satisfied| Satisfied| Satisfied              |
Table A.2: Exponential packet generation data

| Node ID | Exponential distribution | Uniform distribution |
|---------|--------------------------|----------------------|
| t=1     | t=2                      | t=3                  | t=4 | t=5 | t=1 | t=2 | t=3 | t=4 | t=5 |
| 1       | 3.63 2.93 3.41 2.59 2.28 | 6.06 7.55 4.44 2.26 3.1  |
| 2       | 2.37 2.88 3.18 2.27 2.42 | 5.86 6.91 2.36 3.43 2.93 |
| 3       | 3.18 2.28 2.45 3.38 2.54 | 2.41 3.59 7.47 4.18 0.41 |
| 4       | 4.84 2.29 2.52 4.21 4.18 | 2.54 4.03 8.9 8.86 8.74 |
| 5       | 4.07 3.18 3.22 2.33 2.86 | 8.35 7.04 2.78 8.81 5.45 |
| 6       | 2.42 3.1 2.31 2.57 7.68 | 7 2.64 4.35 9.97 6.35 |
| 7       | 2.57 7.61 2.57 7 2.55 | 9.96 4.34 9.93 4.24 9.61 |
| 8       | 5.23 2.38 2.94 3.46 2.66 | 3.17 6.1 7.67 4.84 3.56 |
| 9       | 2.44 3.34 2.49 3.79 3.19 | 7.37 3.85 8.32 6.97 2.53 |
| 10      | 2.29 2.51 4.11 3.9 3.41 | 3.09 8.79 8.5 7.56 4.47 |
| 11      | 2.59 2.27 2.41 3.14 7.91 | 2.34 3.39 6.8 3.97 4.37 |
| 12      | 2.58 2.23 2.27 2.43 3.25 | 2.04 2.38 3.49 7.13 3.07 |
| 13      | 2.37 2.86 3.09 4.83 2.29 | 5.78 6.63 9.41 5.24 4.93 |
| 14      | 2.51 4.17 5.05 3.75 3.13 | 8.86 8.72 8.26 6.76 9.86 |
| 15      | 6.27 2.51 4.14 3.98 3.57 | 4.01 8.83 8.62 7.93 5.67 |
| 16      | 2.84 2.99 3.78 3.19 2.29 | 8.29 8.32 6.96 2.3 3.9 |
| 17      | 2.49 3.88 3.37 2.53 4.45 | 8.47 7.45 4.1 9.14 3.65 |
| 18      | 5.54 2.4 3.05 4.36 4.76 | 3.3 6.52 9.06 9.37 2.4 |
| 19      | 2.27 2.44 3.33 2.48 3.76 | 3.56 7.57 3.84 8.28 8.44 |
| 20      | 3.15 2.24 2.31 2.58 2.24 | 2.12 2.96 4.41 2.17 2.8 |
| 21      | 2.33 2.67 2.46 3.5 2.73 | 4.88 3.68 7.77 5.17 4.65 |
| 22      | 2.62 2.35 2.75 2.7 2.54 | 2.92 5.29 5.92 4.15 9.31 |
| 23      | 4.68 2.25 2.35 2.76 2.72 | 2.24 2.94 5.52 7.14 4.86 |
| 24      | 2.61 2.31 2.57 6.63 2.54 | 2.63 4.33 9.9 4.14 9.28 |
| 25      | 4.63 2.24 2.3 2.56 5.97 | 2.11 2.63 4.31 9.81 3.85 |
| 26      | 2.49 3.78 3.18 2.27 2.42 | 8.31 6.92 2.37 3.46 7.04 |
| 27      | 3.22 2.33 2.66 2.43 3.27 | 2.78 4.82 3.5 7.18 3.22 |
| 28      | 2.39 2.98 3.7 3.04 4.22 | 8.25 8.17 6.47 8.91 8.69 |
| 29      | 4.2 4.15 4.01 3.65 2.96 | 8.84 8.66 8.08 6.17 7.9 |
| 30      | 3.56 2.82 2.93 3.39 2.56 | 5.6 6.04 7.5 4.27 6.69 |
| 31      | 5.48 2.42 3.21 3.21 2.59 | 3.45 7.01 2.67 4.46 2.32 |
| 32      | 2.26 2.4 3.07 4.49 5.7 | 3.31 6.55 9.17 7.95 6.65 |
| 33      | 2.45 3.46 2.66 2.43 3.28 | 7.67 4.83 5.51 7.22 3.34 |
| 34      | 2.41 3.1 5.04 2.36 2.73 | 8.66 9.52 2.9 7.12 4.74 |
| 35      | 2.64 2.39 2.99 3.75 3.13 | 3.23 6.27 8.26 6.76 9.6 |
| 36      | 6.3 2.51 4.18 4.09 3.85 | 4.02 8.87 8.77 8.42 7.29 |
| 37      | 3.3 2.44 3.36 2.52 4.27 | 3.58 7.43 4.05 8.97 0.07 |
| 38      | 4.38 4.86 2.3 2.55 4.99 | 9.43 2.6 4.21 9.5 2.81 |
| 39      | 2.33 2.68 2.48 2.66 2.96 | 4.91 3.79 8.11 9.26 8.22 |
| 40      | 3.71 3.99 4.86 2.3 2.54 | 6.63 9.43 2.58 9.17 9.35 |
| 41      | 4.75 2.27 2.42 3.21 2.31 | 2.36 3.45 7.01 2.68 4.47 |
| 42      | 2.59 2.27 2.42 1.2 2.3 | 2.36 3.44 6.98 2.57 4.13 |
| 43      | 2.53 4.55 6.79 2.54 4.87 | 9.22 9.92 4.19 9.43 2.61 |
| 44      | 2.3 2.55 5.21 2.38 2.94 | 4.24 9.6 3.15 6.01 7.41 |
| 45      | 3.55 2.51 4.04 3.71 3.85 | 3.56 8.59 8.18 8.52 9.05 |
| Node ID | Exponential distribution | Uniform distribution |
|---------|--------------------------|----------------------|
|         | t=1 | t=2 | t=3 | t=4 | t=5 | t=1 | t=2 | t=3 | t=4 | t=5 |
| 46      | 4.35 | 4.74 | 2.27 | 2.41 | 3.12 | 9.35 | 2.34 | 3.37 | 6.71 | 9.78 |
| 47      | 5.8 | 2.47 | 3.57 | 2.84 | 2.99 | 3.73 | 7.95 | 2.67 | 2.37 | 6.95 |
| 48      | 3.74 | 5.12 | 5.71 | 2.46 | 3.47 | 6.73 | 9.75 | 3.66 | 7.69 | 6.5 |
| 49      | 2.67 | 2.47 | 3.62 | 2.91 | 3.31 | 3.76 | 8.02 | 5.98 | 7.31 | 3.64 |
| 50      | 2.45 | 3.44 | 2.63 | 2.38 | 2.8 | 7.62 | 4.67 | 2.99 | 5.51 | 5.76 |
| 51      | 2.86 | 3.07 | 4.56 | 7.02 | 2.55 | 6.57 | 9.23 | 6.93 | 4.25 | 9.61 |
| 52      | 5.26 | 2.39 | 2.97 | 5.6 | 2.88 | 3.2 | 6.19 | 7.98 | 5.86 | 8.91 |
| 53      | 3.17 | 2.26 | 2.41 | 3.08 | 4.77 | 2.33 | 3.53 | 6.92 | 3.38 | 2.42 |
| 54      | 2.28 | 2.45 | 3.42 | 2.61 | 2.3 | 3.63 | 7.58 | 4.55 | 2.62 | 2.27 |
| 55      | 2.56 | 5.51 | 4.34 | 2.43 | 2.98 | 9.7 | 3.48 | 7.11 | 3.01 | 5.56 |
| 56      | 2.81 | 2.9 | 3.24 | 2.36 | 2.8 | 5.92 | 7.11 | 2.99 | 5.51 | 5.76 |
| 57      | 2.86 | 3.07 | 4.6 | 2.23 | 2.27 | 6.58 | 9.26 | 2.04 | 2.38 | 3.5 |
| 58      | 2.43 | 3.27 | 2.39 | 2.99 | 3.76 | 7.18 | 3.23 | 6.28 | 8.28 | 8.44 |
| 59      | 3.15 | 2.24 | 2.5 | 2.54 | 4.7 | 2.1 | 2.58 | 4.16 | 9.33 | 2.26 |
| 60      | 2.26 | 2.37 | 2.89 | 2.31 | 2.32 | 3.11 | 5.9 | 7.03 | 2.74 | 6.67 |
| 61      | 2.03 | 2.36 | 2.82 | 2.92 | 3.36 | 3.02 | 5.59 | 6.02 | 7.43 | 4.03 |
| 62      | 2.52 | 4.21 | 4.19 | 4.15 | 3.94 | 8.91 | 8.88 | 8.81 | 8.57 | 7.76 |
| 63      | 3.5 | 2.72 | 2.6 | 2.29 | 2.52 | 5.14 | 4.53 | 2.54 | 4.03 | 8.9 |
| 64      | 4.21 | 4.18 | 4.1 | 3.85 | 3.32 | 8.87 | 8.77 | 8.43 | 7.33 | 7.31 |
| 65      | 2.46 | 3.55 | 2.8 | 2.85 | 3.82 | 7.87 | 5.5 | 5.71 | 6.42 | 8.74 |
| 66      | 4.07 | 3.8 | 3.21 | 3.32 | 2.62 | 8.34 | 7.03 | 2.72 | 4.61 | 2.8 |
| 67      | 2.53 | 2.67 | 2.47 | 3.57 | 2.82 | 4.89 | 3.73 | 7.91 | 5.62 | 6.1 |
| 68      | 2.94 | 3.47 | 2.67 | 2.47 | 3.64 | 7.69 | 4.91 | 3.77 | 8.07 | 6.13 |
| 69      | 2.95 | 3.51 | 2.74 | 2.65 | 2.4 | 7.78 | 5.21 | 4.76 | 3.29 | 6.47 |
| 70      | 3.04 | 4.22 | 4.2 | 4.16 | 4.62 | 8.51 | 8.9 | 8.84 | 8.88 | 8.14 |
| 71      | 3.68 | 3.01 | 3.94 | 3.49 | 2.71 | 6.37 | 8.56 | 7.75 | 5.08 | 4.34 |
| 72      | 2.57 | 6.69 | 2.54 | 4.72 | 2.26 | 9.91 | 4.16 | 9.34 | 2.31 | 5.28 |
| 73      | 2.4 | 3.03 | 4.13 | 3.93 | 3.48 | 6.44 | 8.81 | 8.56 | 7.73 | 5.03 |
| 74      | 2.7 | 2.54 | 4.89 | 2.31 | 2.54 | 4.19 | 9.45 | 2.65 | 4.38 | 2.05 |
| 75      | 2.23 | 2.28 | 2.46 | 3.5 | 2.73 | 2.44 | 3.68 | 7.78 | 5.18 | 4.67 |
| 76      | 2.03 | 2.36 | 2.81 | 3.68 | 3.15 | 2 | 5.53 | 2.84 | 5.83 | 2.08 |
| 77      | 2.23 | 2.28 | 4.48 | 3.68 | 3.81 | 2.47 | 3.8 | 8.14 | 6.36 | 8.55 |
| 78      | 3.93 | 3.48 | 2.7 | 2.53 | 4.62 | 7.73 | 5.02 | 4.14 | 9.27 | 2.08 |
| 79      | 2.23 | 2.29 | 2.51 | 4.1 | 3.85 | 2.53 | 3.99 | 8.77 | 8.43 | 7.33 |
| 80      | 3.32 | 2.46 | 3.53 | 2.78 | 2.78 | 3.71 | 7.84 | 5.4 | 5.4 | 5.4 |