Hindawi
Journal of Sensors
Volume 2019, Article ID 7409329, 22 pages
https://doi.org/10.1155/2019/7409329

Research Article

On the Use of Graphs for Node Connectivity in Wireless Sensor Networks for Hostile Environments

Emmanuel García-González,[1] Juan C. Chimal-Eguía,[1] Mario E. Rivero-Angeles,[2] and Vicent Pla[3]

[1]*Simulation and Modeling Laboratory, CIC-Instituto Politécnico Nacional, Mexico City, Mexico*
[2]*Network and Data Science Laboratory, CIC-Instituto Politécnico Nacional, Mexico City, Mexico*
[3]*Universitat Politècnica de València, Valencia, Spain*

Correspondence should be addressed to Mario E. Rivero-Angeles; mriveroa@ipn.mx

Received 9 May 2019; Accepted 26 July 2019; Published 19 November 2019

Academic Editor: Jesús Lozano

Wireless sensor networks (WSNs) have been extensively studied in the literature. However, in hostile environments where node connectivity is severely compromised, the system performance can be greatly affected. In this work, we consider such a hostile environment where sensor nodes cannot directly communicate to some neighboring nodes. Building on this, we propose a distributed data gathering scheme where data packets are stored in different nodes throughout the network instead to considering a single sink node. As such, if nodes are destroyed or damaged, some information can still be retrieved. To evaluate the performance of the system, we consider the properties of different graphs that describe the connections among nodes. It is shown that the degree distribution of the graph has an important impact on the performance of the system. A teletraffic analysis is developed to study the average buffer size and average packet delay. To this end, we propose a *reference node* approach, which entails an approximation for the mathematical modeling of these networks that effectively simplifies the analysis and approximates the overall performance of the system.

1. Introduction

Wireless sensor networks are deployed to monitor specific physical variables for many applications, such as animal tracking in forests, structural health monitoring in buildings, or even ambulatory medical surveillance in body area networks. However, in some hostile environments, sensor nodes are placed in adverse situations where they cannot directly communicate to neighboring nodes due to obstacles, interference, noise, or even cyberattacks based on denial of service techniques that shadow some specific connection among nodes. Based on this fact, the remaining topology (connection among nodes) of the network becomes relevant. In this work, we study the performance of the system based on the specific topology of the network by using graph theory. Graphs can be described by many properties such as degree distribution (D. D.), defined as the probability distribution of a node degree over the whole network; clustering coefficient (C. C.), defined as the measure of the degree to which nodes in a graph tend to cluster together; density, defined as the fraction of existing edges in the graph compared to a complete graph; average distance (A. D.), defined as the average number of steps along the shortest paths for all possible pairs of nodes in the graph; and diameter, which is the longest of all shortest paths. From these, the degree distribution is of great interest. Indeed, this is a property that captures, to a large extent, the essence of the form of the graphs that share the same type of distribution.

Graph properties have been used before in the literature to study the performance of the system. For instance, in [1],

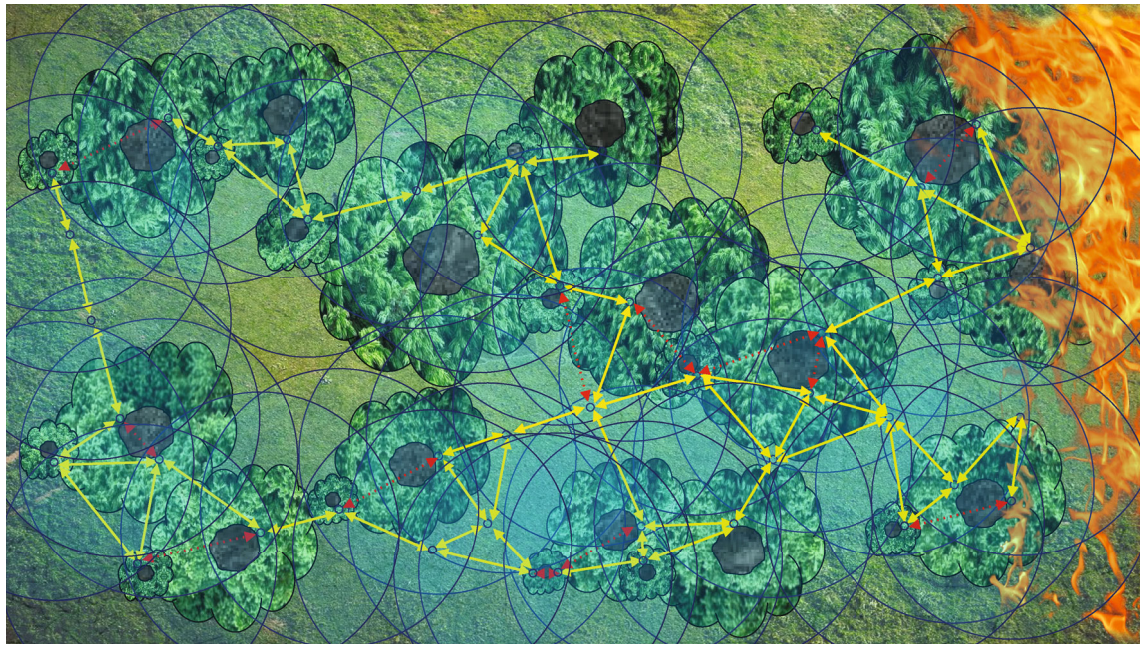


Figure 1: Hostile environment network graph. Yellow arrows represent actual network links, while red dashed arrows represent broken links due to environment conditions.

the author uses two types of graphs, random geometric and clustered graphs, in order to select the position of sensor nodes. Also, in the area of node localization, in [2], the estimation of node's localization in WSNs is proposed using the concept of rigid graphs. However, in these works, the impact of the graph's properties on the average buffer size and packet delay has not been studied. Graph theory is used in [3] to reduce interference in resource allocation schemes using the maximal independent set concept of graphs. However, in this work, the topology of the system is not considered since only small star networks are studied.

Additionally, in such hostile environments, sensor nodes are prone to suffer damage which also reduces data gathering efficiency [4–6]. Furthermore, when interference levels are high or in case of a jamming attack [7], nodes cannot communicate to a sink node that may be found outside the monitored area. If emergency personnel is relying on retrieving information from nodes in the system, this becomes a major issue since some nodes may be disconnected from the sink node and cannot assist the police, military personnel, or firefighters in their specific operation. Hence, we propose sensor nodes to send information regarding the particular environment to other nodes in the system instead of a single sink node as commonly proposed in WSNs. As such, data is not sent to a single sensor node that may be out of reach from many nodes or even destroyed. Conversely, we consider a scenario where sensors send their information to other nodes of interest in a specific area selected before the installation of the WSN. For instance, the network administrator may be interested in relaying information from the northern surveilled area to the eastern area, or from one particular floor of a building to another relevant floor. In this way, the emergency personnel that passes close to the sensor nodes can retrieve relevant information regarding the conditions on different zones of the system.

For instance, consider the network depicted in Figure 1. In this case, nodes are deployed in a forestal fire to assist the personnel in rescue operations. Due to obstacles and the fire itself, nodes can only connect to a few other nodes in the network. These connections can be described by an undirected graph with the following characteristics: (a) it must be a simple graph, i.e., a pair of nodes can be connected by at most one edge and (b) it must be connected. By knowing the properties of this graph, the network administrator can know beforehand the performance of the system in terms of average buffer size and average packet delay. Hence, they can decide if more nodes are needed in order to enhance the monitoring capabilities or even place a few nodes in very particular areas to change or improve the underlying graph and consequently reduce packet delay. Conversely, when a sensor node can communicate with all neighbor nodes, i.e., there are no physical impediments (obstacles, interference, or cyberattacks) for nodes to directly send packets to any other node in its communication range, a random geometric graph or a unit disk graph can accurately describe the topology of the network. In Figure 2, we present such case. This graph has the following characteristics: it is the intersection graph of a set of unit disks in the Euclidean plane, i.e., it is a graph with one vertex for each disk in the set, and there is an edge between two vertices as long as the corresponding vertices lie within a unit distance of each other.

Performance of WSNs in hostile environments has been studied before in the literature. For instance, in [4, 5], nodes may found themselves disconnected from the rest of the network due to such hostile environments that cause unreliable links. To address this issue, the authors propose the use of

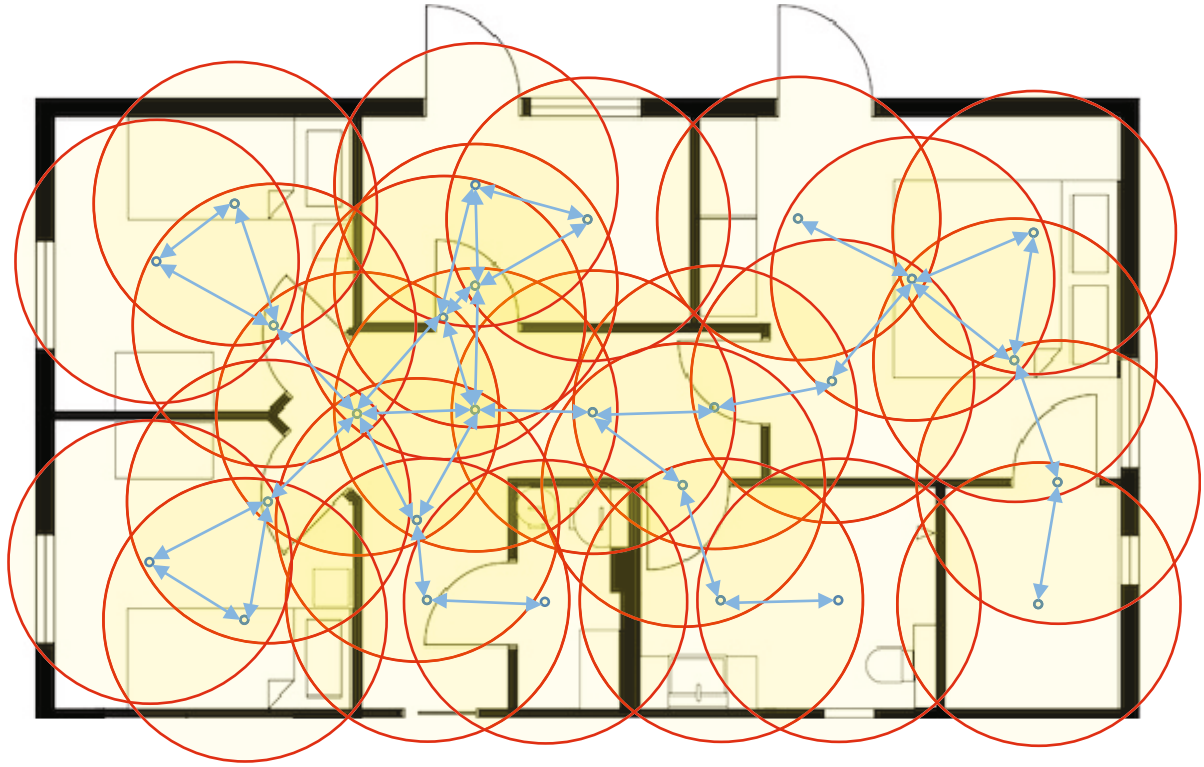


Figure 2: Connectivity among nodes in a unit disk graph topology.

mobile data carriers to temporarily provide connectivity to the rest of the nodes. Unlike this work, we do not consider the case where nodes are disconnected, i.e., we assume that all nodes remain connected either directly or indirectly to all other nodes in the system. Hence, we do not focus on the problem of reconnecting parts of the network. In [6], unreliable links are modeled using an on/off process and the author studies the topology of the network related to nodes degree. Unlike this work, we consider unreliable links to be of no practical use throughout the operation of the system. Additionally, [6] does not consider graph theory to infer the performance of the system in terms of average buffer size nor packet delay. Finally, in [7], links are unreliable due to direct jamming cyberattacks. The authors propose a statistical approach to detect such attacks that may lead to disconnected segments in the system. Unlike this work, we do not focus on detecting such attacks since we assume that operations in hostile environments are only for short operation times and the WSNs are used for assistance in the work field. Note that cyberattack detection requires a considerable amount of time; hence, we believe that there is no real gain on detecting these DoS (denial of service) attacks for these applications since the rescue operation may be over before the attack is detected. The main contributions of this paper are as follows:

(i) A WSN in hostile environments is studied, evaluated, and analyzed. In this case, the hostility is twofold: for one part, it prevents nodes from connecting to neighbor nodes, and secondly, since nodes are prone to malfunction or even destruction, gathered information is disseminated throughout the network instead of a single sink node

(ii) We use the properties of the graph that describe the topology of the WSN in order to determine if the performance of the system is adequate for different system conditions. As such, the network administrator can use a particular graph (by changing the node's position or the number of nodes) to achieve a target performance metric

(iii) An approximate mathematical analysis is developed to study the WSN is such a hostile condition, which can also be used in other applications with similar connectivity properties

The rest of the paper is organized as follows: in Section 2, we present the main assumptions and implications of the studied system. Then, Section 3 details the graph generation process and main properties. The mathematical analysis is developed in Section 4 based on a discrete time Markov chain (DTMC). Finally, the paper concludes presenting relevant numerical results and conclusions.

2. System Model

We now describe the main assumptions and parameters considered in this work. Also, the general operation of the system is presented.

Nodes are uniformly placed in the monitored area and they are connected according to a specific topology to their neighbor nodes considering that many direct wireless links are not reliable, i.e., there are many obstacles, interference, and/or noise in the path between two neighbor nodes (nodes geographically close to each other such that they are in the theoretical communication range) but they cannot directly communicate among them. As such, only some connections are practical among nodes in the network. These connections in the system can be described by graphs with specific characteristics.

Nodes do not transmit directly to a sink node, as *conventional* WSNs. Rather, packets containing information from the monitored phenomena are transmitted to other nodes in the system. Indeed, since we are considering the deployment of nodes in hostile environments, we propose to have redundancy in case of node's malfunction or destruction. Additionally, this data dissemination mechanism allows faster reporting data to be available in the area of interest, for example, in forestal fires or tactical military operations where the personnel is constantly moving inside the monitored area and requires information regarding certain conditions in other parts of the tactical operation. As such, a firefighter crossing the northern area connects directly to a node placed at that zone and can have relevant information from the eastern area without directly connecting to a sink node that may be out of reach. Building on this, nodes generate new packets with probability ρ to be conveyed to specific nodes selected in advance, depending on the system conditions. These packets are routed using the shortest path routing protocol to convey data from source nodes to destination nodes passing through intermediate nodes using the available topology. On the other hand, packets are transmitted with probability τ.

From this description, we can now have a general idea of the system operation. Once that nodes have been deployed in the hostile environment, the network administrator chooses the pair of nodes to be connected, either directly or by multihops, to each other for the duration of the operation, like a fire, hostage situation, and industrial monitoring. At this point, the system operation parameters can be carefully selected by considering the number of nodes in the network and the specific graph described by the available connection among nodes. Specifically, by observing the topology of the nodes, the degree of the graph can be calculated and its probability distribution can be inferred. Then, the values of ρ and τ can be finely selected using the mathematical methodology developed in the following sections in order to calculate average buffer size and packet delay for all of these parameters. As such, the network operation can be known in advanced and it can be improved if needed by adding additional nodes or even changing the placement of some nodes to vary the topology of the graph.

The aforementioned analysis considers a *reference* node in order to simplify the complexity. Indeed, in a WSN where multiple destination nodes are considered and multihop transmissions are allowed, traffic at different nodes may be very different, as traffic conditions depend on the routing protocol and the network topology. By considering a single reference node, we can focus on only one buffer behavior. The reference node, however, can represent the worst case scenario, since we consider that this reference node is one of the nodes used as relay nodes (intermediate node). Our mathematical analysis is an approximation of the real system conditions in the sense that we assume that neighbor nodes to this reference node have similar traffic conditions and, consequently, similar packet arrival probabilities. The validity of these assumptions and the accuracy of the approximation are verified comparing to system simulation results.

3. Graph Generation

In this section, we explain in detail the graph generation process that describes the connections among nodes in the network. Specifically, we focus on generating simple and connected graphs (also called strict graph [8] which is an unweighted, undirected graph containing no graph loops or multiple edges [9, 10]) with a certain degree distribution. Recall that for a WSN with unreliable links, we are interested on investigating the impact of the graph's properties that describe node's connections, on the performance of the system.

3.1. Graphical Degree Sequence Generation. To obtain degree sequences (a list of nonnegative integers that for each vertex of the graph states how many neighbors it has) of simple connected graphs that also follow a certain degree distribution, we first define the number of nodes of the graph, also called the sequence of the graph. After this, we follow the next steps: (a) pseudorandom number generation, (b) discretization of generated number sequence, (c) checking the simplicity of the degree sequence, (d) potentially connected graphic sequences, and (e) forcibly 1-connected graphic sequences. We now detail each of these steps.

In order to study the effect of different probability distributions on the degree of the graph, the following distributions were selected:

(i) Binomial distribution

(ii) Exponential distribution

(iii) Extreme values distribution

(iv) Normal distribution

(v) Power law distribution

(vi) Discrete uniform distribution

These random numbers are generated using the native libraries in C++ language with the only exception of the power law distribution. For this case, the probability density function (pdf) used is the one described in [11].

Now, for the discretization of generated number sequence, we simply choose either the *round* or the *ceil* functions. For specific parameters of some distributions, we might get values below 0.5, which would become 0 if we only use the *round* function. As seen in the next subsection, this would lead us to nonconnected graph. For these cases, we

use the *ceil* function. At this point, we have a sequence of integers of length n.

Then, the simplicity of the degree sequence is verified. As mentioned before, it is an important requirement that the sequence of integers obtained in the previous step does form a simple graph. To achieve this, we assure that the random sequence complies with the conditions of the Havel-Hakimi theorem [12, 13] as follows:

Theorem 1. *Let $S = (d_1, d_2, \cdots, d_n)$ be a finite list of nonnegative integers in nonincreasing order.*

(1) The list S is graphic if and only if the sum of d_i is even, $d_1 > |S| - 1$, and the finite list $S^0 = (d_2 - 1, d_3 - 1, \cdots, d_{d1+1} - 1, d_{d1+2}, \cdots, d_n)$ has nonnegative integers and is graphic

(2) The list is sorted in nonincreasing order on each iteration if necessary

(3) Steps 1 and 2 will be applied at most $n - 1$ times, setting in each further step $S = S^0$

(4) If at any point the list has negative numbers or $d_1 > |S| - 1$, the theorem proves that the list S from the beginning is not graphic

(5) Otherwise, if the whole list S^0 consists of zeros, then the list S from the beginning is graphic and the process ends

Theorem 1 only guarantees that the sequence belongs to a simple graph, but now we verify if it is connected. A potentially connected degree sequence S is that which, of all graphs whose degree sequence is S, at least one of them is connected. In other words, if we want to build a graph from S, we have the certainty that there exists a configuration in which the graph is connected. In order to check if there is a simple connected graph with the sequence of integers we have so far, on Theorem 2 [14].

Theorem 2. *Let $S = (d_1, d_2, \cdots, d_n)$ be a finite list of nonnegative integers in nonincreasing order, $n \geq 2$. A necessary and sufficient condition for the existence of the simple connected graph G with degree sequence S is that*

$$d_n \geq 1. \tag{1}$$

$$\sum_{i=1}^{n} d_i \geq 2(n-1). \tag{2}$$

$$\sum_{i=1}^{k} d_i \text{ is even}. \tag{3}$$

$$\sum_{i=1}^{k} d_i \leq \sum_{i=1}^{k} \bar{d}_i \quad (k = 1, 2, \cdots, n). \tag{4}$$

Since in [14], it is stated that (3) and (4) guarantee the existence of a simple graph with degree sequence S, which is equivalent to the Havel-Hakimi algorithm; we only check if S fulfills the conditions of (1) and (2).

```
   Input : A graph G(V,E)
   Output: True if G is connected, False otherwise
1  begin
2      found ⟵ number of nodes found by BFS
         algorithm ran over G.
3      if found == |V| then
4          return True
5      end
6      else
7          return False
8      end
9  end
```

Algorithm 1: Check connectedness of a graph.

Algorithm 3 in the Appendix implements both Theorem 1 and 2 conditions.

Additionally, we implement an algorithm that checks if a given graph degree sequence is forcibly 1-connected. A forcibly 1-connected sequence S is such that every graph whose degree sequence is S is 1-connected, i.e., it does not matter how the nodes are connected with each other, as long as the rule of *no loops or multiple edges* is observed. Hence, the graph has its nodes connected at most by one edge and there is a path between any pair of nodes.

To this end, we use an algorithm based on Theorem 3 taken from [15] which defines a *sufficient condition* for a degree sequence to be forcibly n-connected. As we are only interested on simple graphs, we use $n = 1$.

It is important to note that this is only a *sufficient condition*, and therefore, the fact that a degree sequence does not fulfill the condition does not necessarily mean that such sequence is not forcibly connected. The full procedure to check if a graphic sequence is forcibly connected is detailed in Algorithm 4 in the Appendix.

3.2. Building Simple Graphs with Given Degree Sequence. At this point, we already have a degree sequence that we know describes a simple graph, either potentially connected or forcibly connected. The next step is to build the graph from that sequence. To achieve this, we consider the Havel-Hakimi algorithm that we now describe.

This algorithm builds a graph using the method described in Theorem 1 but adding some extra steps that provide randomness (this can be seen in lines 2 and 20, where the shuffle in line 2 increases the chances for the degrees to be assigned to different nodes on each run depicted in lines 4-8). Algorithm 5 details the steps needed to build graphs using this method. The purpose of shuffling in line 20 has a deeper reason. Sometimes, when the sequence provided as input is potentially connected, the output graph may result disconnected, and if line 20 is not added, it would always be the same, since the sort in line 21 does not change the relative position of elements when their values are the same [16]. To increase the chances of getting a connected

```
   Input : A simple connected graph G(V,E), τ, ρ, maxSlots
   Output: Statistics of network's performance
1  begin
2     Define if there will be a refNode and which of the
        v_i's will be.
3     while slotCount < maxSlots do
4        foreach v_i ∈ V do
5           p ⟵ random(0, 1)
6           if v_i = refNode and p ≤ ρ then
7              Choose a node t ∈ V uniformly at
                random as destination; t = v_i
8              c_{v_i,t} ⟵ Shortest path from v_i to t
9              Queue a new pkt in v_i's buffer with route
                c_{v_i,t}
10          end
11       end
12       foreach v_i ∈ V do
13          p ⟵ random(0, 1)
14          if v_i has packets in its buffer p ≤ τ then
15             Attempt to transmit the next pkt in v_i's
                buffer
16          end
17       end
18       foreach v_i ∈ V do
19          Check status of every pkt received in v_i and
             do counting.
20       end
21       slotCount++
22    end
23    Calculate Pa^1, Pa^+, P(Q = 0), E[Q]
24    Calculate E[D_e]
25 end
```

Algorithm 2: General operation of the simulator.

graph, line 20 is added, opening possibilities of getting a different graph in multiple runs of the algorithm.

So far, we can produce a graph by one of the previously mentioned methods. However, we have to check if such graph is indeed connected when using the forcibly connected degree sequences or the Chung-Lu method. To this end, we follow the procedure presented in Algorithm 1, where we first run a breadth-first search (BFS) algorithm over the graph we want to test. If BFS can reach every node in the graph, then it is connected.

In this section, we described the specific methods for generating the graphs that describe the connections among the nodes in the system. Such connections imply a direct path between any pair of nodes. We are now interested on analyzing the performance of the sensor network when nodes are connected in such manner. To this end, an analytical model is derived in the following section.

4. The Mathematical Model

In this section, we describe in detail the mathematical analysis based on a DTMC to model the average buffer length and packet delay. To this end, we propose the use of a reference node, which is a *conventional* node, i.e., a node in no extraordinary conditions. Like the majority of the nodes in the system, such conventional node may not be placed in the edges of the monitored area and it is not isolated from the rest of the nodes. Rather, it would have an average number of neighbor nodes and its nodes are likely to have similar traffic conditions than this reference node. The reason for this consideration is that traffic in WSNs where packets are no longer directed to a single sink node greatly depends on the network topology (given by the graphs described above) and routing algorithm, which in turns corresponds to a very complex system that could be described by a higher number of variables and states. Conversely, by focusing on only one reference node, with similar conditions to the majority of the nodes in the system, it is possible to reduce such complexity and derive close expressions for average buffer size and packet delay irrespective of the network topology and routing protocol.

Building on this, a slotted Non-Persistent Carrier Sense Multiple Access (NP/CSMA) protocol where packets in the buffer of nodes are transmitted with probability τ in a given time slot. This system can be described by a DTMC where state (Q) represents the number of packets in the buffer at slot t of the reference node with valid state space $\{\Omega_Q : 0 \leq Q\}$ as depicted in Figure 3.

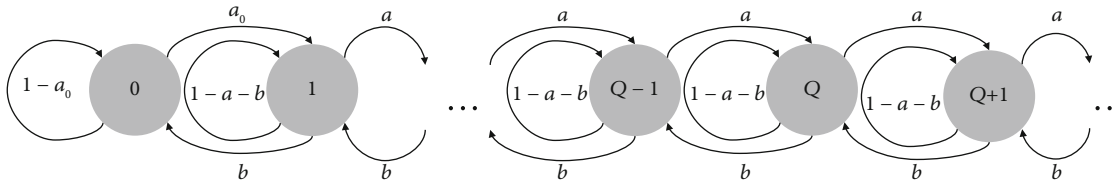


Figure 3: Markov chain state diagram.

Due to the fact that the reference node corresponds to a node with similar traffic conditions than the majority of the nodes and also to its neighbor nodes, we consider that the probability that a single packet arrives to the reference node is $Pa^1(r)$, and the probability that two or more packets arrive to the reference node is given by $Pa^+(r)$. Consequently, the probability that no packet arrives to this node is given by $Pa^0(r) = 1 - Pa^1(r) - Pa^+(r)$.

It is important to remark that this proposed model is an approximation to reduce the complexity of the system since we are also considering that Pa^1, Pa^+, and Pa^0 are also the probabilities of a single, multiple, or no packet arrivals to all the reference node's neighbors, respectively. This is not always the case, since the reference node can be close to nodes in the borders of the surveilled area or can be in a particular environment where many links are not active, which could drastically change the values of packet arrivals in the reference node and neighbor nodes. However, we consider this approximation to provide general and accurate results for WSNs with unreliable links and multiple sink nodes, as proven by comparing the analytical results to extensive simulation results presented further in the paper. To differentiate an event at the reference node and neighbor nodes, we consider the following variables: the probability that a single packet arrives to a neighbor node is $Pa^1(v)$, and the probability that two or more packets arrive to a neighbor node is given by $Pa^+(v)$.

Valid state transition probabilities of the proposed DTMC are as follows:

(i) From state 0 to state 0 with probability $P_{0,0} = \{Pa^1(r)\}$ which corresponds to the case when the buffer is empty and there are no packet arrivals to the reference node

(ii) From state 0 to state 1 with probability $P_{0,1} = \{1 - Pa^1(r)\}$ which corresponds to the case where a single packet is successfully received by the reference node, i.e., no collision occurred at the reference node

(iii) From state Q to state $Q - 1$, $Q > 1$, with probability $P_{Q,Q-1} = \{\tau[1 - P(Q = 0)][1 - Pa^1(v) - Pa^+(v)]\}$. This transition corresponds to the case where a single packet is transmitted by the reference node with probability τ and it is correctly received by the intended neighbor node, i.e., no other packets are transmitted or received by the intended neighbor node. This implies that no other neighbor node transmitted to the intended reference node's neighbor node

(iv) From state Q to state $Q - 1$, $Q > 1$, with probability $P_{Q,Q} = \{\tau[1 - P(Q = 0)][\tau[1 - P(Q = 0)] + [1 - \tau[1 - P(Q = 0)]][Pa^1(v) + Pa^+(v)]] + [1 - \tau[1 - P(Q = 0)]][1 - Pa^1(r)]\}$. In this case, no new packet is successfully received by the reference node. This occurs due to the following cases: (a) the reference node transmits the packet with probability τ. However, the intended neighbor node also transmitted or does not transmit but other neighbor nodes transmitted to it, causing a packet collision; (b) this state transition can also occur when the reference node does not transmit but no other packet is successfully received by the reference node

(v) From state Q to state $Q + 1$, $Q > 1$, with probability $P_{Q,Q+1} = \{[1 - \tau[1 - P(Q = 0)]][Pa^1(r)]\}$. In this case, a new packet is received by the buffer of the reference node. This happens when the reference node does not transmit, and only one packet is received by this reference node, i.e., no packet collision occurred and only one neighbor node transmitted

Note that in order to transmit a packet, the node's buffer has to be nonempty. Hence, the probability to transmit a packet is always given by $\tau[1 - P(Q = 0)]$.

To derive closed expressions for the performance of the system, we first simplify the previous expressions as follows:

(i) $a_0 \longleftarrow P_{0,1}$

(ii) $1 - a_0 \longleftarrow P_{0,0}$

(iii) $a \longleftarrow P_{Q,Q+1}$

(iv) $b \longleftarrow P_{Q,Q-1}$

(v) $1 - a - b \longleftarrow P_{Q,Q}$

Now, we recursively calculate the stable state probabilities as follows:

$$\begin{aligned} \pi_1 &= \frac{a_o}{b} \cdot \pi_0 ; \\ \pi_2 &= \frac{a}{b} \cdot \pi_1 ; \\ \pi_3 &= \left(\frac{a}{b}\right)^2 \cdot \pi_1 ; \\ &\cdots \\ \pi_i &= \left(\frac{a}{b}\right)^{i-1} \cdot \left(\frac{a_0}{b} \cdot \pi_0\right). \end{aligned} \tag{5}$$

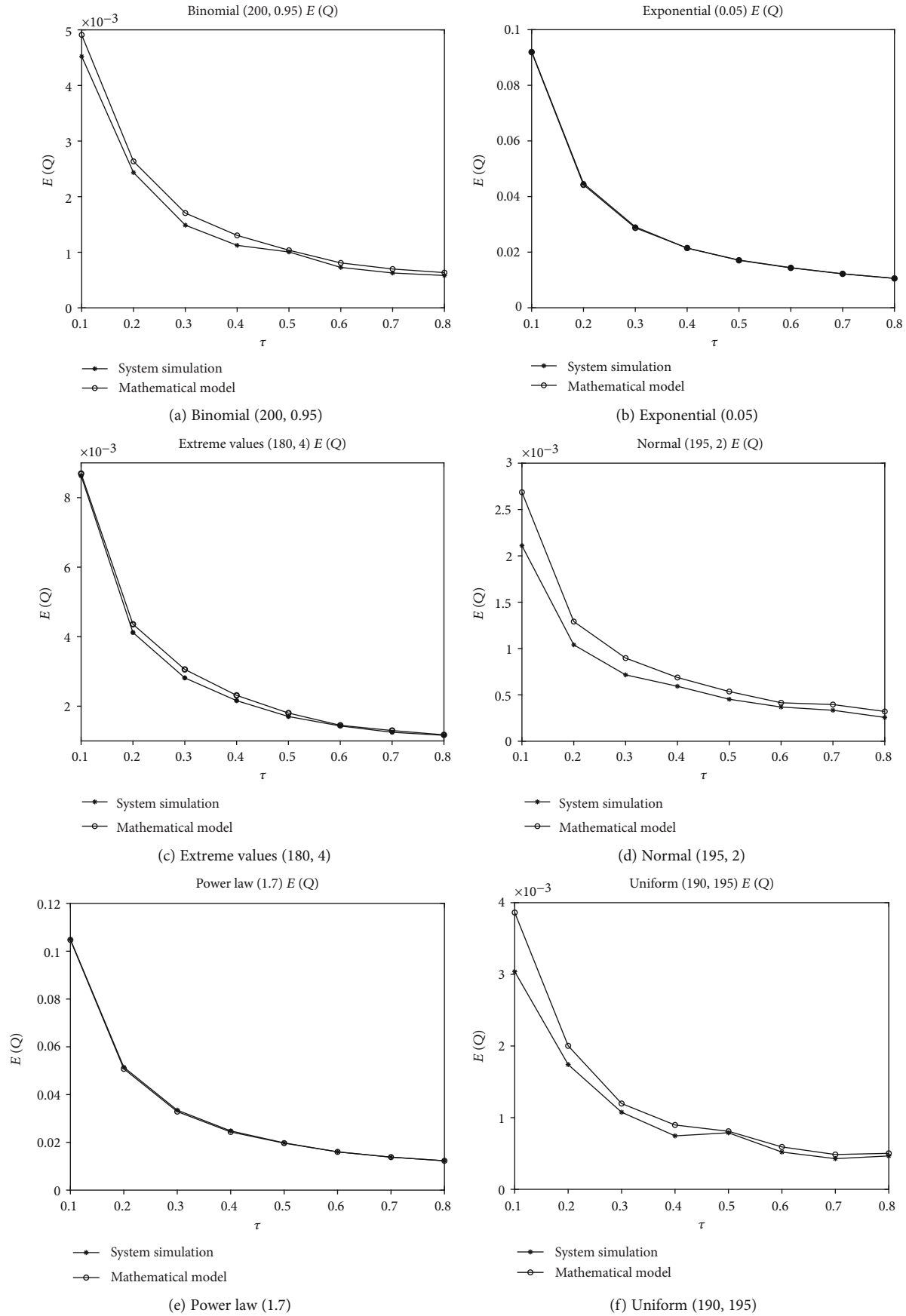


Figure 4: Comparative of $E[Q]$ between the system simulation and DTMC model on networks of 200 nodes.

Table 1: Properties and mean absolute error of graphs with 200 nodes.

Distribution	C.C.	Density	A.D.	Diameter	Mean absolute percentage error
Binomial (200, 0.95)	0.9537	0.953869	1.04613	2	10.2505%
Exponential (0.05)	0.3328	0.0998995	2.25	6	0.5596%
Extreme values (180, 4)	0.9195	0.917136	1.08286	2	4.3269%
Normal (195, 2)	0.9806	0.979799	1.0202	2	20.8797%
Power law (1.7)	0.664	0.0439196	1.98553	3	0.7926%
Uniform (190, 195)	0.9658	0.966884	1.03312	2	13.9472%

And using the normalization equation

$$\sum_{i=0}^{\infty} \pi_i = 1, \tag{6}$$

we get

$$\sum_{i=0}^{\infty} \pi_i = \pi_0 \left[1 + \frac{a_0}{b} + \frac{a_0}{b}\left(\frac{a}{b}\right) + \frac{a_0}{b}\left(\frac{a}{b}\right)^2 + \cdots\right]. \tag{7}$$

From this, after some algebraic manipulation, we can calculate π_0 as follows:

$$\pi_0 = \frac{b-a}{b-a+a_0}. \tag{8}$$

Now, substituting π_0 from (8) in (5) we get

$$\pi_i = \left(\frac{a}{b}\right)^{i-1}\left(\frac{a_0}{b}\cdot\frac{b-a}{b-a+a_0}\right). \tag{9}$$

The average buffer size $E[Q]$ can be calculated as $E[Q] = \sum_{i=0}^{\infty} i\pi_i$. Hence, using (9) and after some algebraic manipulation, we can express $E[Q]$ as follows:

$$E[Q] = \sum_{i=0}^{\infty} i\left(\frac{a}{b}\right)^{i-1}\left(\frac{a_0}{b}\cdot\frac{b-a}{b-a+a_0}\right) = \frac{a_0 b}{(b-a)(b-a+a_0)}. \tag{10}$$

To calculate the average packet delay, we use Little's theorem as follows:

$$E[D] = \frac{E[Q]}{\lambda}, \tag{11}$$

where arrival rate λ packets per unit of time can be described by $\lambda = Pa^1(r)$.

Finally, probabilities Pa^1 and Pa^+ cannot be directly derived since they greatly depend on the network topology and routing protocol. We believe that deriving closed expressions for these probabilities falls outside the scope of this work, and we leave this research area open for future works. However, we obtain these probabilities by means of numerical simulations using a *home-made* network simulator built in C++ described in Algorithm 2.

As mentioned in System Model, new packets are generated with probability ρ and are transmitted to specific destination nodes selected in advanced using many intermediate nodes to reach them. Packets are transmitted between every pair of nodes with probability τ. From this simulation, we can obtain numerical values for the packet arrival probabilities at each node in the system.

5. Numerical Results

In this section, we present the most relevant results that describe the performance of the system based on the properties of the graphs used to define the topology of the network.

We first validate the mathematical model by comparing the analytical results to the system simulation results. In Figure 4, we have shown the comparison between results of average buffer size from the system simulation and the DTMC for networks with 200 nodes and different values of τ. The system parameters used to obtain the numerical results and absolute mean error are shown in Table 1. We can see a very good match for all distributions and different graphs. Specifically, the best approximations occur for the exponential and power law distributions, where the predominant feature is that the C.C. value is much larger than the density.

In order to see if the number of nodes in the network has a relevant impact on the accuracy of the model, we consider the case of networks from 10 to 50 nodes with power law distribution as shown in Figure 5. It can be seen that as the number of nodes in the system increases, there is a lower error between the analytical and simulation results.

So far, we have compared the DTMC results with networks created by our algorithm, which can be considered to some extent to have random topologies. But there are special cases of networks where the conditions of the model are more suited to the conditions of the system. Recall that the main assumption of the mathematical model is that a reference node has similar traffic conditions that its neighbors, in such a way as to consider the packet arrival probabilities to be equal. To this end, we consider a 51-node network with a topology where all links are symmetrical in all directions, so the traffic conditions are fulfilled in the reference node. In Figure 6, we show the graphic representation of these networks. Note that graph 1 is a tree, and the rest of the graphs are some variations of graph 1 with

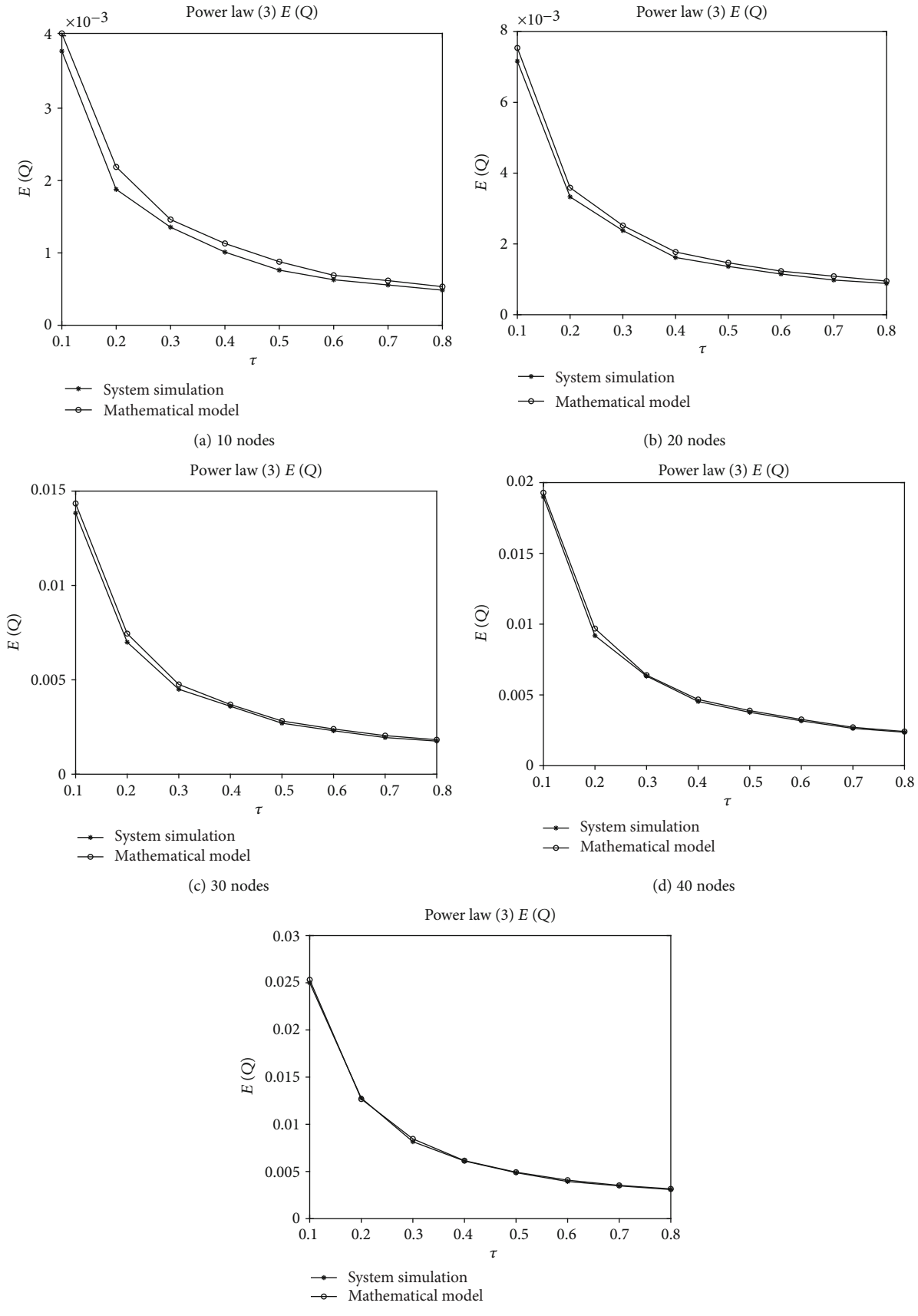
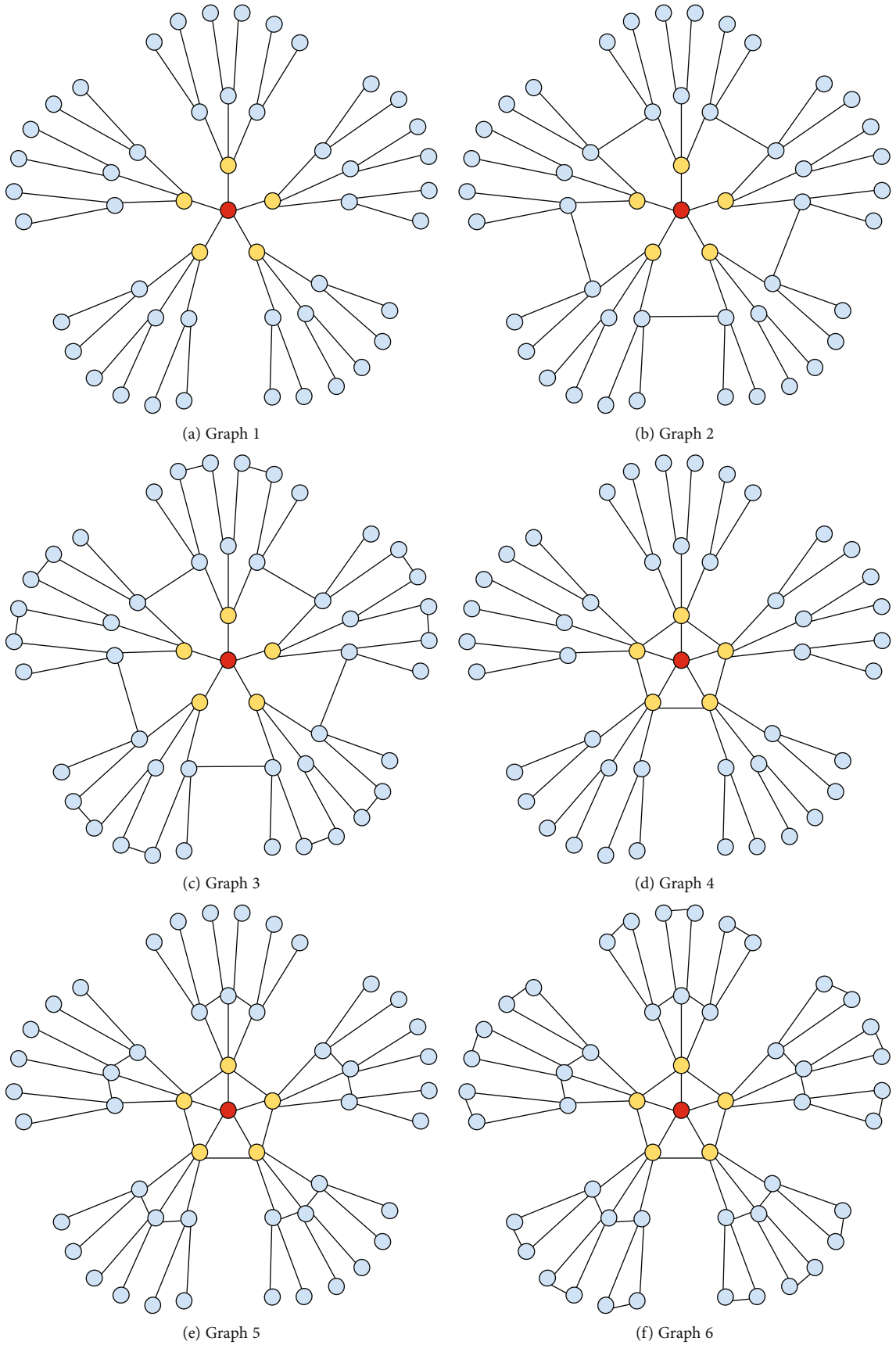


Figure 5: Comparative of $E[Q]$ between the system simulation and DTMC model.



(a) Graph 1

(b) Graph 2

(c) Graph 3

(d) Graph 4

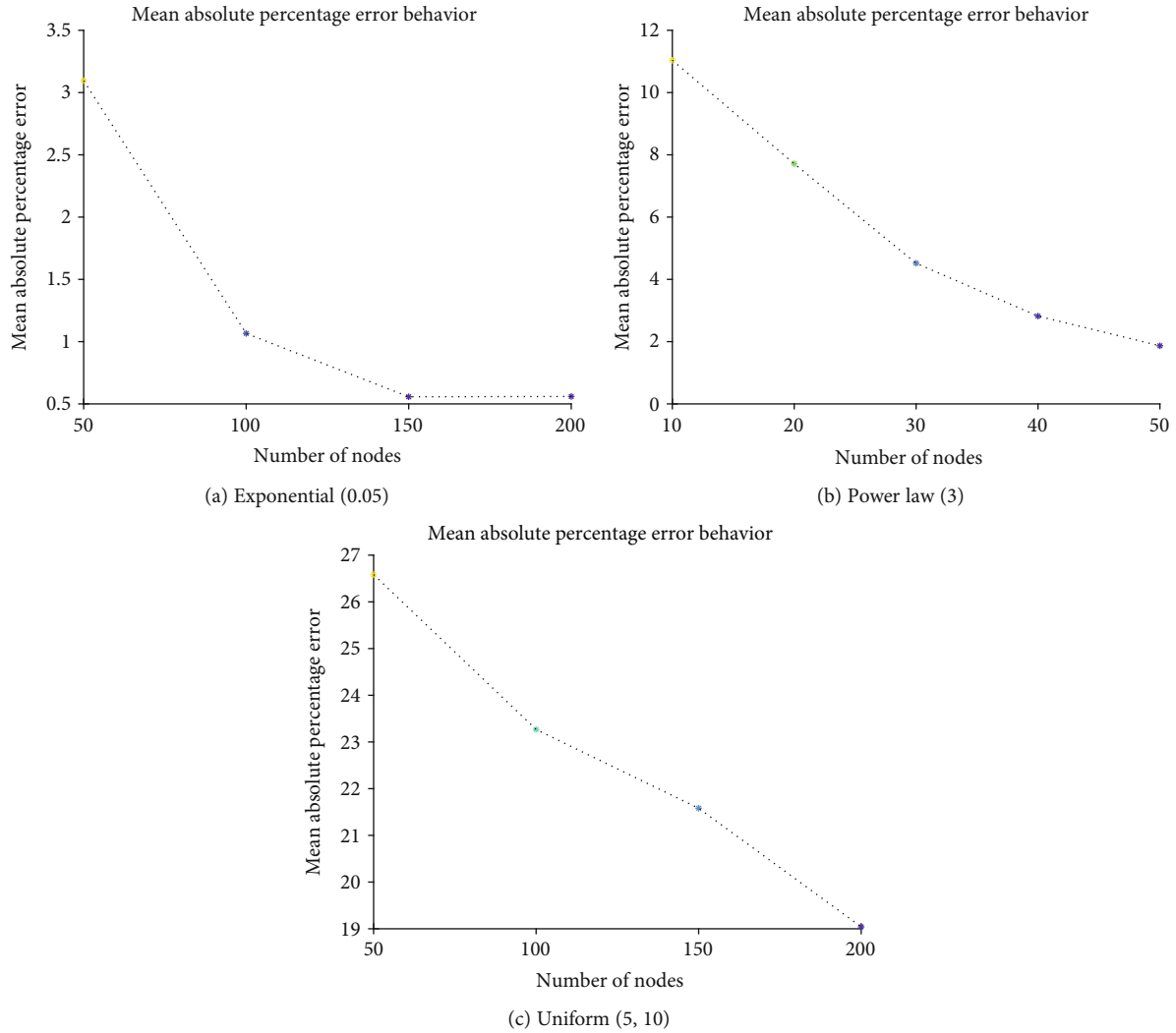
(e) Graph 5

(f) Graph 6

Figure 6: 51 nodes graphs of ideal kind for the mathematical model.

Table 2: Properties and mean absolute error for networks of degree distribution uniform (5, 10).

Graph name	C.C.	Density	A.D.	Diameter	Mean absolute percentage error
Graph 1	0	0.0392157	4.47843	6	1.9978%
Graph 2	0	0.0431373	4.20784	6	3.4965%
Graph 3	0	0.0509804	4.12549	6	2.8714%
Graph 4	0.0222	0.0431373	4.08627	6	5.0145%
Graph 5	0.08834	0.0509804	4.01569	6	3.8470%
Graph 6	0.71935	0.0627451	4.00392	6	5.1753%



(a) Exponential (0.05)

(b) Power law (3)

(c) Uniform (5, 10)

Figure 7: Behavior of mean absolute percentage error from the number of node point of view.

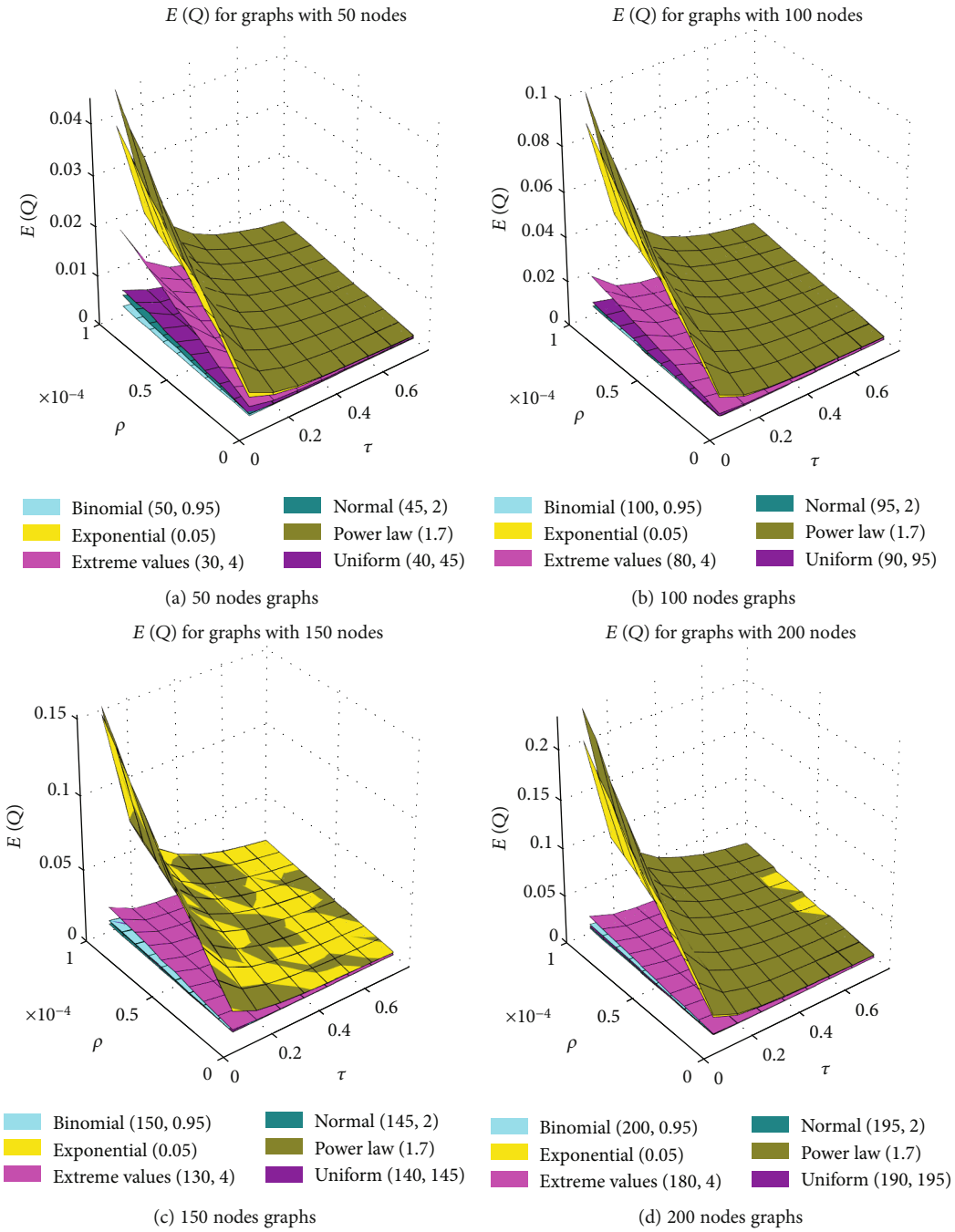
small mutations that change in some extent the graph's properties, as it is defined in Table 2, where we also present the mean absolute percentage error between analytical and simulation results. Specifically, in graphs 1, 2, and 3, the density increases while C.C. remains the same. In this case, the error also increases but it does not exceed 3%. In graph 4, the C.C. is smaller, while in graph 5, the C.C. is higher than the density. In graph 6, the C.C. is quite larger than the density value and the error is above 5%.

Finally, as the number of nodes increases, we can see a better fit between the analytical results and the simulation results for all distributions as depicted in Figure 7.

Now that we validated our mathematical model, we investigate the effects of the properties of graphs on the performance of the system. To this end, we first study

Table 3: Test graphs degree distribution parameters.

Number of nodes	50 nodes	100 nodes	150 nodes	200 nodes
Distribution parameters	Binomial (50, 0.95)	Binomial (100, 0.95)	Binomial (150, 0.95)	Binomial (200, 0.95)
	Exponential (0.05)	Exponential (0.05)	Exponential (0.05)	Exponential (0.05)
	Extreme values (30, 4)	Extreme values (80, 4)	Extreme values (130, 4)	Extreme values (180, 4)
	Normal (45, 2)	Normal (95, 2)	Normal (145, 2)	Normal (195, 2)
	Power law (1.7)	Power law (1.7)	Power law (1.7)	Power law (1.7)
	Uniform (40, 45)	Uniform (90, 95)	Uniform (140, 145)	Uniform (190, 195)



(a) 50 nodes graphs

(b) 100 nodes graphs

(c) 150 nodes graphs

(d) 200 nodes graphs

Figure 8: Comparative of $E[Q]$ between graphs with the same number of nodes.

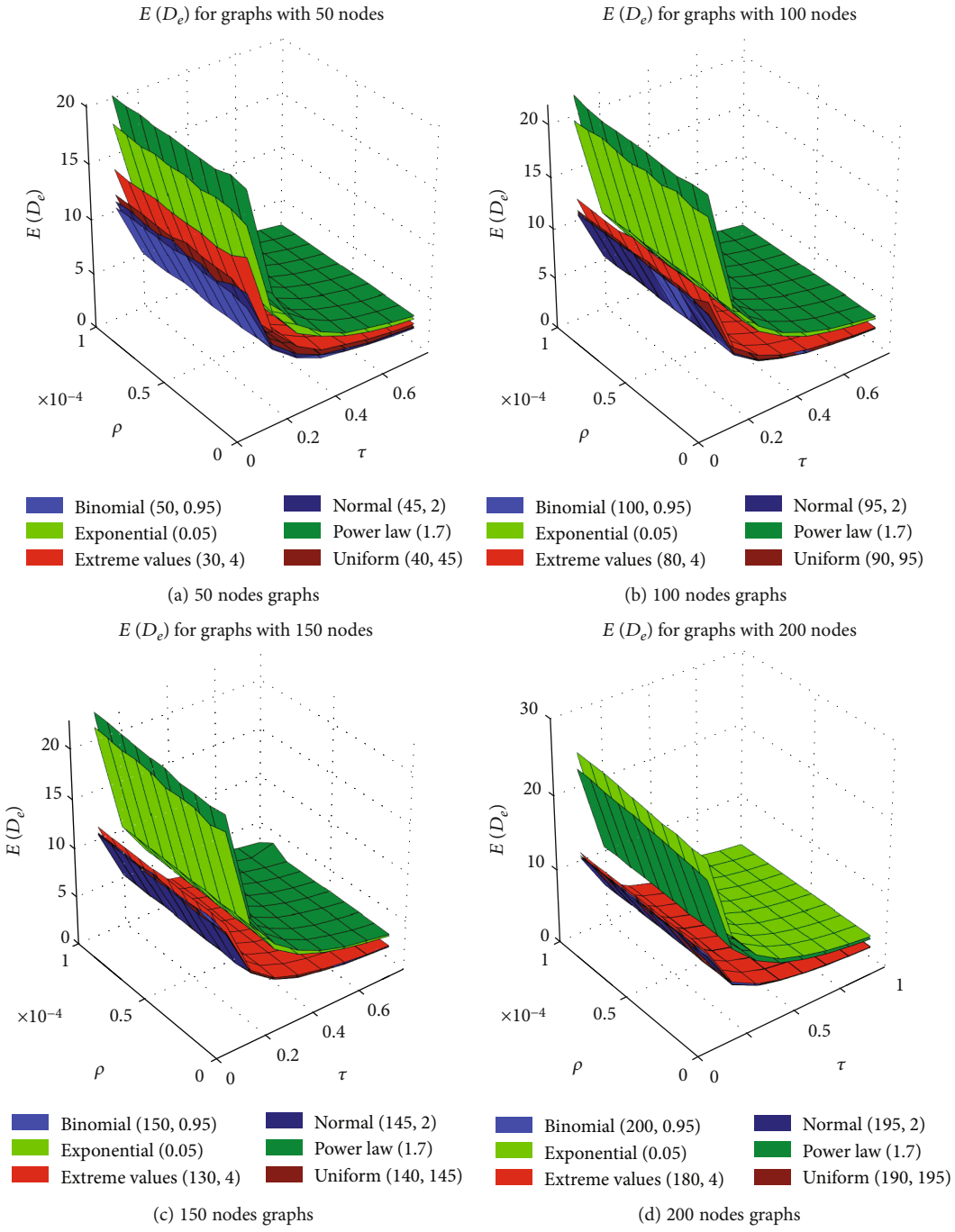


FIGURE 9: Comparative of $E[D_e]$ between graphs with the same number of nodes.

the effect of the different degree distributions for different numbers of nodes in the network. To this end, we consider the different probability distribution functions describe above with their specific parameters for each network density. These parameters and numerical values are presented in Table 3.

These graphs were built from these sequences using the Havel-Hakimi method described in Algorithm 5. The following parameters were used for data packet traffic $0.1 \leq \tau \leq 0.8$ and $1 \times 10^{-5} \leq \rho \leq 1 \times 10^{-4}$. We took a *reference node* as the node present in most of the short paths between every pair of nodes to calculate $E[Q]$. The reason behind choosing the reference node in this way is because every packet generated follows the shortest path to its destination node. Hence, the reference node is one of the *bottlenecks* of the network. In other words, the chosen node can be seen as the "worst case" and we can expect that the rest of the nodes in the network have equal or lower values of their buffer size.

Since a network simulator was developed, we now validate the mathematical approximated model to the simulation results in order to determine the accuracy of the proposed

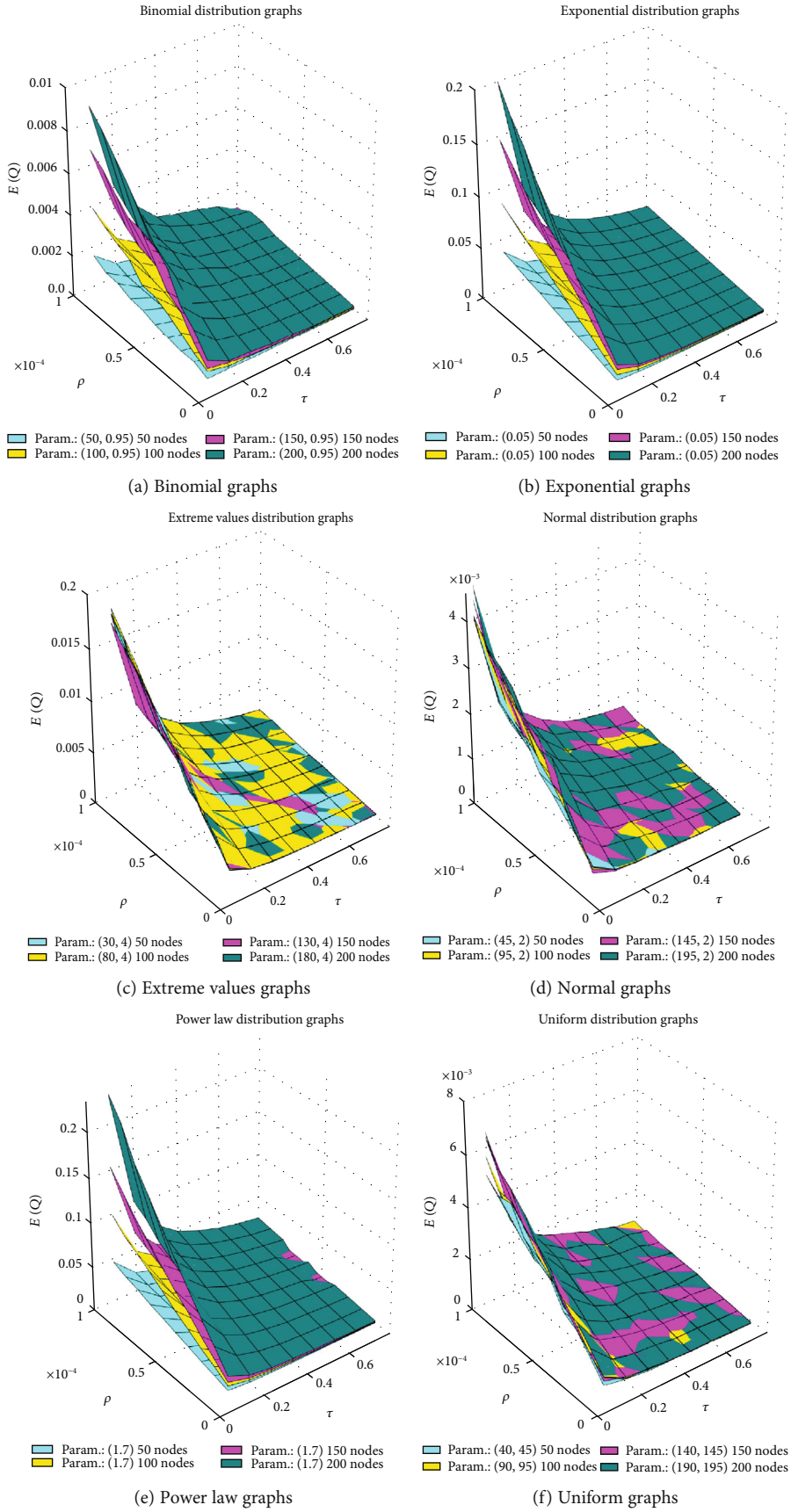
Table 4: Simulation outcomes and properties of graphs used.

# nodes	Degree distribution	Min. $E[Q]$	Max. $E[Q]$	Min. $E[D_e]$	Max. $E[D_e]$	C.C.	Density	A.D.	Diameter
50 nodes	Binomial (50, 0.95)	0.000011	0.001545	1.265287	10.514759	0.9774	0.974694	1.02531	2
	Exponential (0.05)	0.000472	0.037791	2.133920	17.778186	0.7908	0.282449	1.71755	2
	Extreme values (30, 4)	0.000224	0.017376	1.661500	14.223935	0.6952	0.666122	1.33388	2
	Normal (45, 2)	0.000042	0.003875	1.340659	11.135802	0.928	0.925714	1.07429	2
	Power law (1.7)	0.000579	0.044836	2.438283	20.272008	0.6832	0.106939	1.94776	3
	Uniform (40, 45)	0.000066	0.004988	1.419981	11.716424	0.8653	0.866122	1.13388	2
100 nodes	Binomial (100, 0.95)	0.000051	0.003903	1.302432	10.634500	0.9564	0.958788	1.04121	2
	Exponential (0.05)	0.001025	0.085422	2.323515	19.670225	0.5585	0.193939	1.87495	2
	Extreme values (80, 4)	0.000202	0.017992	1.424887	12.046218	0.8273	0.830101	1.1699	2
	Normal (95, 2)	0.000058	0.003936	1.292603	10.533097	0.9624	0.961212	1.03879	2
	Power law (1.7)	0.001235	0.099275	2.555177	21.917932	0.6524	0.0682828	2.04909	4
	Uniform (90, 95)	0.000081	0.005701	1.338307	11.239774	0.9371	0.936364	1.06364	2
150 nodes	Binomial (150, 0.95)	0.000101	0.006693	1.306539	10.841755	0.9535	0.955884	1.04412	2
	Exponential (0.05)	0.001815	0.145864	2.492453	21.259371	0.4343	0.137987	1.97423	5
	Extreme values (130, 4)	0.000260	0.016563	1.407530	11.340467	0.8906	0.889217	1.11078	2
	Normal (145, 2)	0.000049	0.004271	1.278812	10.525010	0.9711	0.972617	1.02738	2
	Power law (1.7)	0.001647	0.152268	2.609807	22.765497	0.7357	0.0519016	2.10631	4
	Uniform (140, 145)	0.000072	0.006639	1.286138	10.630950	0.9554	0.955973	1.04403	2
200 nodes	Binomial (200, 0.95)	0.000118	0.008768	1.326301	10.694603	0.9537	0.953869	1.04613	2
	Exponential (0.05)	0.002327	0.200372	2.852023	24.528760	0.337	0.0998995	2.25	6
	Extreme values (180, 4)	0.000255	0.017480	1.369051	11.104683	0.916	0.917136	1.08286	2
	Normal (195, 2)	0.000064	0.004584	1.286624	10.063900	0.9768	0.979799	1.0202	2
	Power law (1.7)	0.002587	0.232759	2.466938	22.316455	0.664	0.0439196	1.98553	3
	Uniform (190, 195)	0.000091	0.006431	1.293636	10.683731	0.9658	0.966884	1.03312	2

analytical framework. We first compared the values of the average buffer size and average packet delay for all of the degree distributions for graph generation for different numbers of nodes in the network as shown in Figures 8 and 9, respectively. It can be seen that as the density of the network graph is higher, the measurements of the mean buffer size in the reference node and the mean end-to-end packet delay both decrease. Also, a low average buffer size does not necessarily imply a low average packet delay. For instance, consider the case of a network with 100 nodes and low new packet generation probability (low reporting environment) and low transmission probability (low values of τ). In this case, there are just a few packets in the system as the reference node has a low average buffer size. However, these packets can take a long time to be transmitted due to the very low value of τ. As data traffic increases (ρ increases), buffer size always increases but average packet delay does not. From these results, it is important to note that the increase on the data reporting (the number of new packets generated by nodes) does not imply a degradation of the system as long as an adequate value of τ is selected. This fine selection can be done using these results presented in order to achieve a specific average packet delay target.

In Table 4, we show the most relevant values of the graphs used in these experiments, such as minimum and maximum values of $E[Q]$ and $E[D_e]$, and average clustering coefficient, density, average distance, and diameter. These results can be used by the network administrator in order to achieve key performance metrics of the network based on the given topology. Furthermore, if the current topology is not the best suited, it can be modified by adding or moving some nodes. For instance, consider the case when the topology of the current network is described by an exponential distribution when 150 nodes are placed in the hostile environment. In this case, the maximum packet delay would be 21.25 time slots. This value may be higher than desired for the specific service provided by the WSN. Then, the network administrator can replace some nodes in order to have now a normal distribution entailing a 10.52 time slots maximum packet delay.

We also compared all of the graphs that shared the same degree distribution irrespective of the number of nodes in the network in terms of average buffer size and packet delay presented in Figures 10 and 11, respectively. In Figure 10, we can see that the extreme values, normal, and uniform distributions have very similar results, while in Figure 11, we can see a similar tendency in the binomial, normal, and power law distributions. We also noticed that for each network of certain number of nodes, both C.C. and density values are quite close in most cases,and as the number of nodes increases, these values also increase, but the A.D. decreases.



(a) Binomial graphs

(b) Exponential graphs

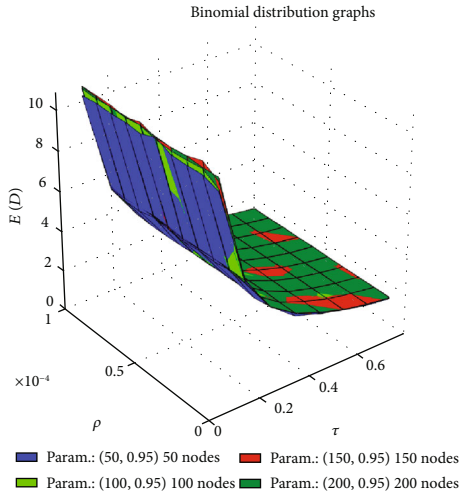
(c) Extreme values graphs

(d) Normal graphs

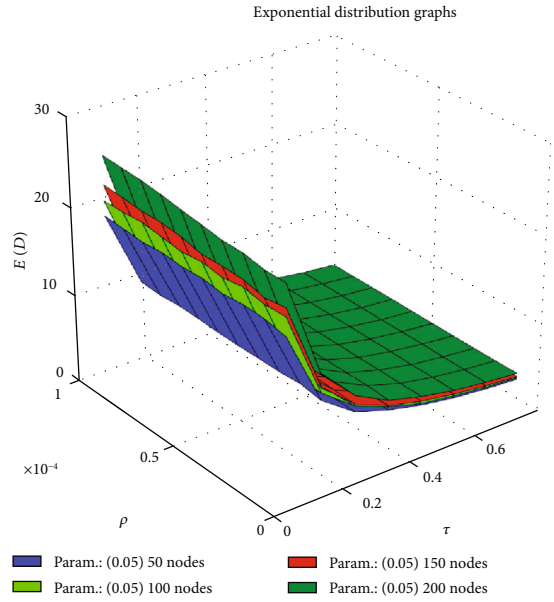
(e) Power law graphs

(f) Uniform graphs

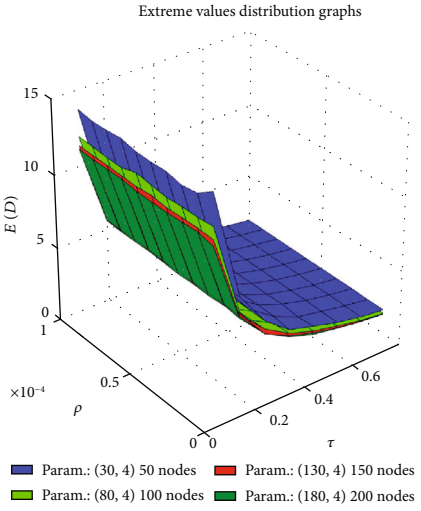
FIGURE 10: Comparative of $E[Q]$ between graphs with the same type of degree distribution.



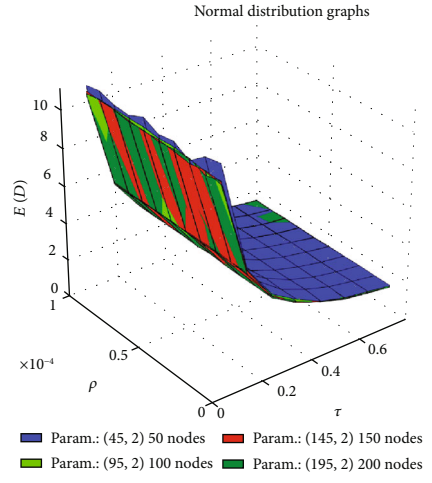
(a) Binomial graphs



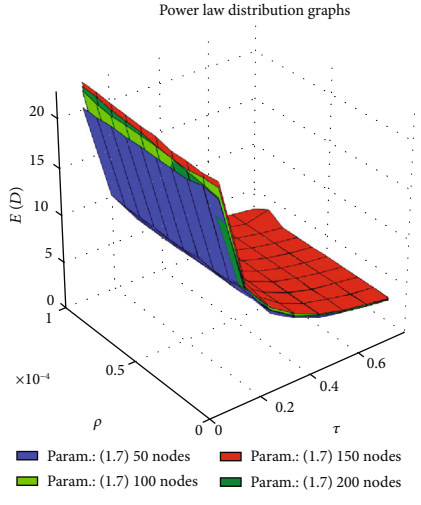
(b) Exponential graphs



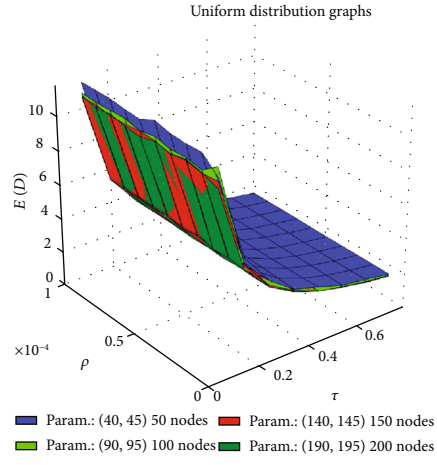
(c) Extreme values graphs



(d) Normal graphs



(e) Power law graphs



(f) Uniform graphs

Figure 11: Comparative of $E[D_e]$ between graphs with the same type of degree distribution.

Table 5: Network graphs with 100 nodes with A.D. values of 2.55.

Distribution	C.C.	Density	A.D.	Diameter
Binomial (35, 0.21)	0.101	0.0769697	2.54869	4
Exponential (0.1)	0.3055	0.107475	2.54646	7
Extreme values (7, 4)	0.1448	0.089697	2.54889	5
Normal (7, 1)	0.0713	0.0715152	2.55475	4
Power law (1.8)	0.4344	0.0567677	2.55556	5
Uniform (1, 20)	0.1453	0.101818	2.55495	7

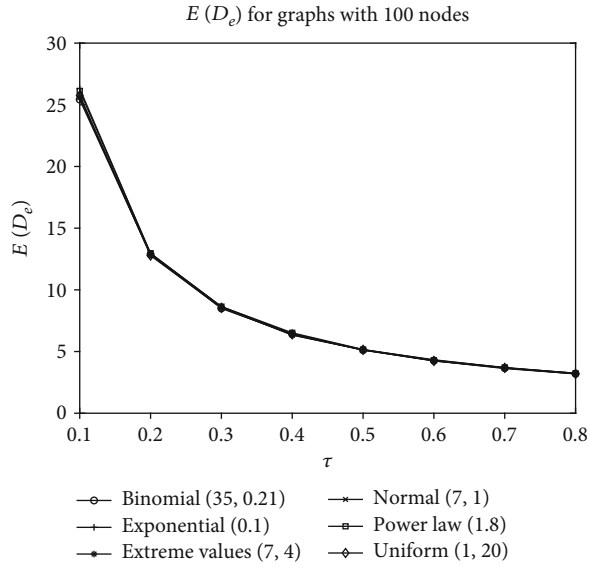


Figure 12: Comparative of $E[De]$ between networks with A.D. of 2.55 approx.

Table 6: Network graphs with 100 nodes with C.C. values of 0.7.

Distribution	C.C.	Density	A.D.	Diameter
Binomial (100, 0.69)	0.70714	0.704848	1.29515	2
Exponential (0.05)	0.71241	0.218586	1.93354	5
Extreme values (65, 5)	0.69599	0.687879	1.31212	2
Normal (70, 15)	0.70733	0.708889	1.29111	2
Power law (1.8)	0.69055	0.0652525	1.96343	3
Uniform (64, 74)	0.6942	0.69697	1.30303	2

Additionally, it is interesting to note that as the number of nodes grow, the value of A.D. remains almost constant. For example, in the binomial distribution, the A.D. values remain mostly in 1.04 approximately and the normal, power law, and uniform distribution graphs also have similar behaviors but it exists a most noticeable difference. As expected, the A.D. values in these distributions have small changes as the number of nodes increases, but they are higher in contrast with the binomial distribution graphs. Now, concerning the exponential distribution graphs, it produces the more variations between different numbers of nodes. For instance, when the network has 50 nodes, the graph has an A.D. of

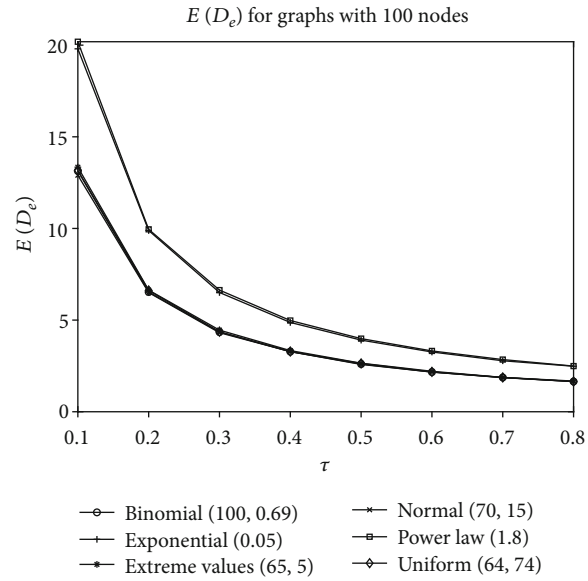


Figure 13: Comparative of $E[De]$ between networks with C.C. of 0.7 approx.

Table 7: Network graphs with 100 nodes with density values of 0.2.

Distribution	C.C.	Density	A.D.	Diameter
Binomial (100, 0.2)	0.21879	0.200202	1.82808	3
Exponential (0.05)	0.5642	0.205051	1.97172	4
Extreme values (18, 4)	0.2354	0.200404	1.83434	3
Normal (20, 10)	0.2039	0.2	1.81677	3
Uniform (16, 24)	0.207	0.20101	1.81596	3

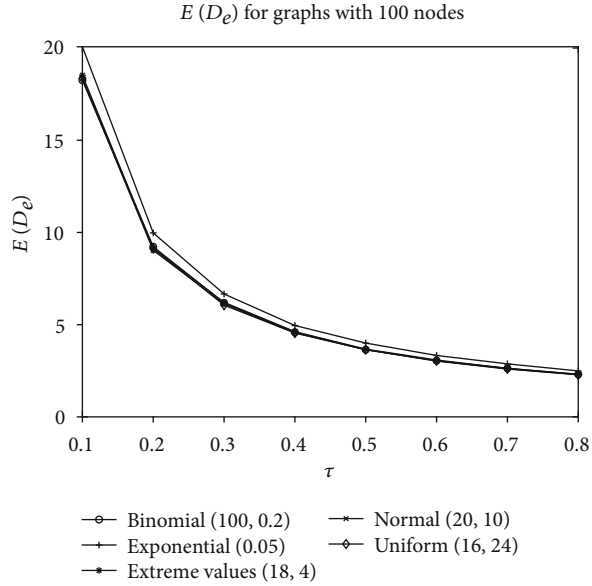


Figure 14: Comparative of $E[D_e]$ between networks with density of 0.2 approx.

```
Input: A list of integers L
Output: True if the L is potentially connected, False
        otherwise
1  begin
2      n ⟵ |L|
3      Sort L in non-increasing order
4      if L_n ≤ 0 then
5          return False
6      end
7      if ∑_{i=1}^{n} L_i ≤ 2(n − 1) then
8          return False
9      end
10     if L_1 ≤ n − 1 then
11         continue ⟵ True
12     end
13     else
14         continue ⟵ False
15     end
16     integrity ⟵ False
17     while continue do
18         actualValue = L_1
19         Delete L_1 from L
20         for i=1 : i≤actualValue do
21             L_i ⟵ L_i − 1
22             if L_i < 0 then
23                 continue ⟵ False
24                 break
25             end
26             else
27                 continue ⟵ True
28             end
29         end
30         if There's some L_i ∈ L > 0 then
31             Sort L in non-increasing order
32         end
33         else
34             continue ⟵ False
35             break
36         end
37     end
38     if integrity then
39         for i = 1: i ≤ n do
40             if L_i ≥ 0 then
41                 result ⟵ True
42             end
43             else
44                 result ⟵ False
45                 break
46             end
47         end
48     end
49     if result then
50         return True
51     end
52     else
53         return False
54     end
55 end
```

Algorithm 3: Check if a list of integers L is a graphical potentially connected degree sequence.

```
Input: A potentially connected degree sequence S
Output: True if S is forcibly 1-connected, False
        otherwise
1  begin
2    Sort S in non-increasing order
3    n⟵1
4    p ⟵ |S|
5    k_initial ⟵ n + 2
6    for k = k_initial : k ≤ p do
7      if S_k ≥ p − k + n then
8        res ⟵ True
9      end
10     else
11       res ⟵ False
12       break
13     end
14   end
15   return res
16 end
```

Algorithm 4: Check if a graphic potentially connected degree sequence S is also forcibly connected.

```
Input: A simple potentially/forcibly connected degree
       sequence D={d_0,d_1,...,d_{N−1}}
Output: A simple graph G(V,E) whose degree
        sequence is D
1  begin
2    Shuffle D
3    V⟵Ø
4    for i = 0 : i ≤ N − 1 do
5      v_i.availableDegree ⟵ d_i
6      v_i.actualDegree ⟵0
7      V⟵V ∪ { v_i}
8    end
9    Sort V in non-increasing order according to their
       availableDedgree.
10   E⟵Ø
11   while v_o. availableDegree
12     k ⟵ v_o.availableDegree
13     u ⟵ v_0
14     for i = 1 : i ≤ k do
15       v ⟵ v_i
16       E⟵E ∪ { u, v }
17       u.availableDegree −−
18       v.availableDegree −−
19     end
20     Shuffle V
21     Sort nodes in V in non-increasing order
         according to its availableDegree.
22   end
23 end
```

Algorithm 5: Building graphs based on Havel-Hakimi.

1.71, while for the case of 100 nodes, the graph has an A.D. of 1.87, and with 150 nodes, the graph has A.D. of 1.97, and in the 200 nodes case, the graph has A.D. of 2.25.

Now, we observe numerical results using graphs with the same A.D. In these results, we use a fixed value of $\rho = 5 \times 10^{-5}$. Specifically, we created a set of networks whose A.D. is approximately 2.55 and compared their values of average packet delay. The properties of the network graphs are detailed in Table 5, and in Figure 12, we show the comparative plot. It can be seen that, when A.D. remains constant, even if the values of C.C. and density are different, the performance of the system is not affected.

We are now interested on using graphs with the same values of C.C. to see the impact on the system's performance. To this end, we created a set of networks whose C.C. is approximately 0.7 and compared their values of average packet delay. The properties of the network graphs are detailed in Table 6, and in Figure 13, we show the comparative plot. In this case, the power law and exponential distributions entail higher average packet delays, while the rest of distributions achieve a better performance. The rationale behind this is that both power law and exponential distributions have similar A.D. values, while the rest of the considered distributions also have similar values of A.D. From this, we can infer that as the A.D. increases, also packet delay increases.

One more set of networks whose density is approximately 0.2 was created and then we show the system performance. The properties of the network graphs are detailed in Table 7, and in Figure 14, we show the comparative plot. In this tests, we have omitted the power law distribution since it was not possible to create a graph of this degree distribution with density of 0.2. The reason of this is related to the probability distribution properties that do not allow to create

dense graphs. In this case, we confirm our previous observation that when graphs have similar density values, the average packet delay is not affected even if the rest of parameters vary.

6. Conclusions

In this work, we consider a WSN in hostile environments where many links between neighbor nodes are unreliable and, hence, cannot be used to establish direct communication among them. Based on this environment, we propose to map the topology of the network to specific graphs with different characteristics such as cluster coefficient, degree distribution, A.D., and diameter. We propose to use these properties to evaluate the performance of the system beforehand in terms of average buffer size and average packet delay by using an approximation methodology that greatly simplifies the analytical framework. We validate the accuracy of the approximation by comparing to extensive simulation results showing an absolute error lower than 5% for most results.

From the different graphs used to represent the topology of the network, we can see that the only property of its graph that clearly affects the system performance in terms of average buffer size and packet delay are the values of the A.D. degree distribution does not have a relevant role in the performance of the system. However, not all values of A.D. can be achieved with all of the considered distributions. For example, with exponential and power law distributions, we can achieve a minimum A.D. value much larger than the ones we can achieve with the rest of the distributions. This is because the first two belong to graphs less dense than the ones we can create with the rest of the distributions.

In future works, we will focus on developing closed expressions for the packet arrival probabilities to provide closed expressions on the average end-to-end packet delay.

Appendix

In this appendix, we present some useful algorithms used throughout the paper. First, we present the theorem that gives a sufficient condition for a graphical sequence to be forcibly n-connected. After this, Algorithm 3 is used to know whether a list of integers is a graphical potentially connected degree sequence. Then, we detail Algorithm 4 used to know if a known-to-be potentially connected degree sequence is also forcibly connected. Finally, we show Algorithm 5 (Havel-Hakimi) used to build graphs from a graphical degree sequence.

Theorem 3. *Let $n \geq 1$ be an integer. Then a graphic sequence $\pi : d_1 \geq d_2 \geq \cdots \geq d_p$ is forcibly n−connected if*

$$d_k \geq p - k + n, \quad \text{for every } k, k \geq d_n + 2. \tag{12}$$

Data Availability

Data supporting the results can be accessed and shared as required by contacting the authors.

Conflicts of Interest

The authors have no conflict of interest either financial or personal that affects the objectivity of the results.

Acknowledgments

The authors wish to thank the Consejo Nacional de Ciencia y Tecnología (CONACyT), the Comisión de Operación y Fomento de Actividades Académicas, Instituto Politécnico Nacional (COFAA-IPN, project numbers 20196225 and 20196678), and the Estímulos al Desempeño de los Investigadores del Instituto Politécnico Nacional (EDI-IPN) for the support given for this work. The work of V. Pla was supported by Grant PGC2018-094151-B-I00 (MCIU/AEI/FEDER, UE).

References

[1] T. Eren, "The effects of random geometric graph structure and clustering on localizability of sensor networks," *International Journal of Distributed Sensor Networks*, vol. 13, no. 12, 2017.

[2] S. Rai B and S. Varma, "An algorithmic approach to wireless sensor networks localization using rigid graphs," *Journal of Sensors*, vol. 2016, Article ID 3986321, 11 pages, 2016.

[3] J. Zhou, L. Wang, W. Wang, and Q. Zhou, "Efficient graph-based resource allocation scheme using maximal independent set for randomly-deployed small star networks," *Sensors*, vol. 17, no. 11, article 2553, 2017.

[4] W. Lalouani, M. Younis, and N. Badache, "Interconnecting isolated network segments through intermittent links," *Journal of Network and Computer Applications*, vol. 108, pp. 53–63, 2018.

[5] S. Lee, M. Younis, B. Anglin, and M. Lee, "LEEF: latency and energy efficient federation of disjoint wireless sensor segments," *Ad Hoc Networks*, vol. 71, pp. 88–103, 2018.

[6] J. Zhao, "Topological properties of secure wireless sensor networks under the q-composite key predistribution scheme with unreliable links," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1789–1802, 2017.

[7] O. Osanaiye, A. S. Alfa, and G. P. Hancke, "A statistical approach to detect jamming attacks in wireless sensor networks," *Sensors*, vol. 18, no. 6, article 1691, 2018.

[8] W. T. Tuutle, *Graph Theory as I Have Known It*, Oxford University Press, Oxford England, 1988.

[9] I. N. Bronshtein, K. A. Sevendyaveu, G. Musiol, and H. Muehlig, *Handbook of Mathematics*, Springer-Verlag, Berlin, Heidelberg, 4th edition, 2004.

[10] A. Gibbons, *Algorithm Graph Theory*, Cambridge University Press, Cambridge England, 1985.

[11] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," *Society for Industrial and Applied Mathematics*, vol. 51, no. 4, pp. 661–703, 2009.

[12] V. Havel, "A remark on the existence of finite graphs," *Časopis Pro Pěstování Matematiky (in Czech)*, vol. 80, pp. 477–480, 1955.

[13] S. L. Hakimi, "On realizability of a set of integers as degrees of the vertices of a linear graph. I," *Journal of the Society for Industrial and Applied Mathematics*, vol. 10, no. 3, pp. 496–506, 1962.

[14] C. Berge, *Graphs and hypergraphs*, North Holland Publishing Company, Amsterdam, 1973, 117-118.

[15] S. A. Choudum, "On forcibly connected graphic sequences," *Discrete Mathematics*, vol. 96, no. 3, pp. 175–181, 1991.

[16] "cplusplus.com, Sort algorithm," November 2017, http://www.cplusplus.com/reference/algorithm/sort/.