Received September 20, 2021, accepted October 15, 2021, date of publication November 2, 2021,
date of current version November 22, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3124737

Energy-Efficient and Delay Sensitive Routing Paths Using Mobility Prediction in Mobile WSN: Mathematical Optimization, Markov Chains, and Deep Learning Approaches

GERMÁN A. MONTOYA, CARLOS LOZANO-GARZON, (Senior Member, IEEE), AND YEZID DONOSO, (Senior Member, IEEE)
Systems and Computing Engineering Department, Universidad de los Andes, Bogotá 111711, Colombia

Corresponding author: Germán A. Montoya (ga.montoya44@uniandes.edu.co)

This work was supported by the Vice Presidency for Research and Creation Publication fund at the Universidad de los Andes, Bogotá, Colombia.

ABSTRACT In Mobile Wireless Sensor Networks there could be scenarios where absolutely all network nodes (including the base station) are mobile, becoming a very hard task to find a communication path between a sensor node and the base station due to many network variables are changing at each moment. In addition, there are delay-sensitive applications that require establishing communication paths as soon as possible to mitigate low network performance in terms of end-to-end delay, reducing, at the same time, the energy consumption of the network. For this reason, we propose a multiobjective mathematical optimization model for finding the optimal communication path between a source node and a sink (base station) considering hard scenarios where all network nodes are mobile and minimizing end-to-end delay and energy consumption. This mathematical model would offer significant advantages to evaluate new algorithms due to we could know how far or close are the algorithm results from the optimal values given by the mathematical model. In addition, we propose a prediction distributed routing algorithm based on Markov Chains that takes into account the network mobility in order to find as fast as possible a communication path between a source node and a sink with minimal energy consumption. We also propose a deep learning approach to predict future nodes' distances in a mobile network to determine if future movements of nodes will cause communication disruptions in paths. Significant findings were obtained when the Markov Chains and Deep Learning approaches were compared in terms of predicting nodes mobility and reducing the delay and the energy consumption in the network. The performance of our prediction algorithms (Markov Chains and Deep Learning approaches) is evaluated against the mathematical model to determine how good it is. Finally, to analyze our prediction algorithms considering real online scenarios, we compared it against typical routing algorithms, obtaining promising results in terms of delay and energy consumption in all mobile node scenarios.

INDEX TERMS Mathematical optimization model for time-varying graphs, delays, energy consumption, prediction algorithm, Markov chains, deep learning.

I. INTRODUCTION

Mobile Wireless Sensor Networks (MWSN) are a particular case of WSN at which some or all network nodes are mobile due to they are attached to entities such as mobile objects, animals, or humans. Due to these entities are continually moving in a specific area, it is probably to experiment long delays to establish a path to communicate a sensor node with a sink (base station), affecting the end-to-end delay performance in delay sensitive applications such as military or healthcare monitoring applications [3]. In addition, these sensors are limited in terms of energy, whereby it is necessary to accomplish delay requirements and, at the same time, minimizing energy consumption [1], [2]. In this sense, due to a sink should not experiment undesired delays when it receives information collected by sensors, it is necessary to propose

The associate editor coordinating the review of this manuscript and approving it for publication was Juan Liu.

mathematical models and algorithms capable of building communication paths as fast as possible with minimal energy consumption [3], [4].

Given the scenario described above, novel routing algorithms are emerging for solving these mentioned problems such as [11]–[13]. However, they, and many others that will be mentioned in section II-B, do not present a mathematical optimization model and only the sinks are mobile in the network; that is, all network nodes are not mobile. Proposing mathematical optimization models for representing networks at which nodes' position changes across time (time-varying graphs) is not an easy task. However, there are mathematical optimization models for time-varying graphs applied to different technological fields, such as wireless networks, cellular networks, and vehicular networks. Nonetheless, these types of mathematical optimization models are scarce in scenarios where all nodes are completely mobile, and they are not formulated considering the particular requirements of mobile wireless sensor networks such as constraints in terms of energy consumption, computational and memory resources. In this sense, our work pretends to propose a mathematical optimization model for time-varying graphs in the context of a mobile wireless sensor network for finding the optimal communication path between a source node and a sink (destination node) considering all network nodes are mobile and minimizing two objective functions: end-to-end delay and energy consumption. The results obtained by the mathematical optimization model could be compared against algorithms in order to evaluate their performance in terms of delay and energy consumption. In other words, optimal values given by the mathematical model can be reference values to determine how good are the results of a new algorithm proposed in MWSN considering, obviously, the same goals and parameters of the mathematical model. This would give us a great advantage to evaluate new algorithms because we would know how far or close are the algorithm results from the optimal values given by the mathematical model. However, we do not pretend our mathematical model represents all the details involved in the design of an algorithm in MWSN, but the comparison results between the proposed algorithm and the mathematical model could give clues to detect if the algorithm we are proposing is going in the right or wrong direction. On the other hand, we also propose prediction distributed routing algorithms to be compared against the mathematical model in order to evaluate its performance and potential application in real mobile wireless sensor networks scenarios. The first prediction algorithm is based on Markov Chains for considering network mobility with the aim of finding as fast as possible a communication path between a source node and a sink with minimal energy consumption. The second prediction algorithm is based on a Deep Learning approach with the same purpose as the Markov Chains method. Both methods are compared later in Results Section (IV) in order to extract their differences and similarities. Finally, this paper is an extension of a previous work presented in [15]. This extension consists of considering the following improvements:

- In the mathematical model, we considered two objective functions: delay and energy functions to approach delay-sensitive applications with minimal energy consumption. These two functions were taken into account to obtain optimal solutions minimizing both functions considering network mobility.
- In the mobility prediction algorithm based on Markov Chains, we considered not only RSSI levels of nodes but their current energy consumption level to be selected as a forwarding node in order to find a path between a sensor source node and a sink.
- We considered more details for the movement model of sensor nodes, that is, we described precisely the theoretical details of the selected movement model.
- The energy consumption model applied to the sensor nodes is described in detail and implemented with realistic parameters in order to obtain results adjusted to real scenarios.
- The energy consumption model applied to the sensor nodes is described in detail and implemented with realistic parameters in order to obtain results adjusted to real scenarios.
- We propose a new prediction algorithm based on deep learning to select the best forwarding node in order to build a path between a sensor source node and a sink. In addition, a dataset has been proposed to be used by our deep learning approach to select the best forwarding node.

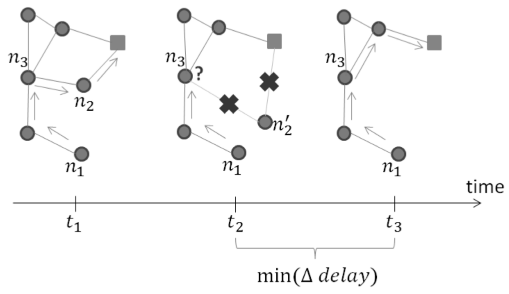
A. PAPER ORGANIZATION

The remainder of this paper is organized as follows. The general problem statement is described in section II-A; the mathematical optimization model for the problem is presented in III-A, and the mobility prediction algorithm based on Markov Chains is introduced in section III-B. In Section III-C, the mobility prediction algorithm based on Deep Learning is described. In section IV we presents the results obtained; and finally, section V shows the conclusions.

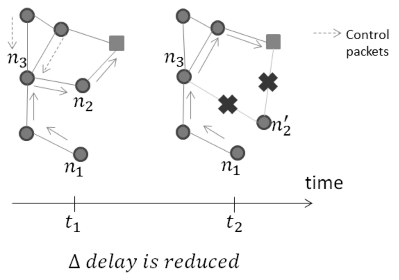
II. PROBLEM STATEMENT AND RELATED WORKS

A. PROBLEM STATEMENT

Figure 1.a illustrates our problem. Suppose we have an MWSN, which at time t_1 there is a communication path between the source node n_1 and a destination node (squared node). However, at time t_2, node n_2 moves away from node n_3, causing a communication disruption for transmitting information from n_1 to the destination node. Once n_3 has realized this problem at time t_3, n_3 has to perform routing corrections in order to reestablish the communication path between n_1 and the destination node. The communication reestablishment of this path can be perfectly performed using routing techniques but at the expense of introducing



(a) Problem



(b) Solution

FIGURE 1. Problem definition.

undesired delays. In some applications, these delays can be ignored because they do not affect application goals, but in other ones, such as delay sensitive applications like health monitoring, this disadvantage might result in a low end-to-end delay network performance.

Given the problem above, our proposal consists of using a prediction technique described in Figure 1.b [7], [8], [18]. It represents the same situation showed in Figure 1.a, but in this case, node n_3 at time t_1 receives information that indicates node n_2 will rapidly be away from its communication range at time t_2. Given this information, n_3 at time t_1 is also analyzing a possible candidate node, which could replace n_2 in the case of n_2 fails at a future time. Indeed, if node n_2 at time t_2 fails because it has moved away from n_3, this node at time t_2 can promptly restore the communication path between n_1 and the destination node, reducing the delay described in Figure 1.a.

Proposing mathematical optimization models for representing networks at which nodes' position changes across time (time-varying graphs) is not an easy task. However, there are many mathematical optimization models for time-varying graphs applied to different technological fields, such as wireless networks, cellular networks, and vehicular networks. Nonetheless, this type of mathematical optimization model is scarce in mobile wireless sensor networks due to this kind of wireless network has particular constraints in terms of energy consumption, computational and memory resources. In this sense, our work pretends to propose a mathematical optimization model for time-varying graphs in the context of a mobile wireless sensor network.

B. RELATED WORKS

Due to we propose three different methods to solve the problem, that is, a mathematical optimization approach, a Markov Chains approach, and a Deep Learning approach, we have divided the related works into three parts respectively, which will be described as follows.

1) MATHEMATICAL OPTIMIZATION RELATED WORKS

Given the problem statement above, novel routing algorithms are emerging for solving these mentioned problems. Specifically, in [11] the authors propose an algorithm to guarantee that each source sensor node gets single hop access to a backbone node in order to reach a mobile sink. However, they do not present a mathematical optimization model and only the sinks are mobile in the network; that is, all network nodes are not mobile.

Authors in [12] present a multi-objective particle swarm optimization for finding the optimal path in a wireless sensor network with a mobile sink for data collection. This proposal corresponds to an evolutionary approach that cannot guarantee the optimal solution and all network nodes are not mobile. In other words, they do not propose a mathematical optimization model in order to obtain optimal solutions.

In [16], the authors present a minimally invasive veneer tree using the particle optimization algorithm for routing wireless sensor networks with a moving sink. This algorithm is population-based, and population members try to find a tree that has less energy and latency by sharing routing information. The proposed algorithm was compared in terms of energy consumption, distance, and the number of steps with previous algorithms. However, this work does not present a mathematical optimization model to obtain optimal values, instead of that, they present a metaheuristic that does not guarantee obtaining optimal values. In addition, they only consider that the sink is mobile and not the rest of sensor nodes.

In [17], the authors propose a mobile sink path planning for collecting the information for static nodes located at the bottom of the sea. They develop a Cluster Head Selection Algorithm (CHSA) and particle swarm optimization algorithm (PSO) to optimize the selection of cluster heads in the Underwater Heterogeneous Sensor Network (UHSN). Their proposed method can balance and save nodes energy consumption while shortening the moving path of the mobile sink. However, this work does not present a mathematical optimization model to obtain optimal values, instead of that, they present a metaheuristic that does not guarantee obtaining optimal values. In addition, they only consider that the sink is mobile and not the rest of sensor nodes.

In [18], the authors propose a Stochastic Optimal Routing Algorithm (SORA) for high-loss WSNs. In SORA, the energy conservation and transmission delay reduction problems are firstly transformed into a stochastic optimization problem. However, the scenario evaluated by them is completely static, that is, none of the network nodes are mobile.

In [19], the authors propose a routing algorithm that has an ant colony optimization (ACO) algorithm. This algorithm uses an endocrine cooperative particle swarm optimization algorithm (ECPSOA) that is used to improve several metrics in WSNs routing such as End-to-end delay, power consumption, and communication cost. The algorithm is evaluated in completely mobile network scenario, whereby is very seemed to our proposal. However, their proposal is a metaheuristic that does not guarantee to obtain optimal values like our mathematical optimization model.

In [20], the authors propose a mobility and energy-aware cross-layer searching routing algorithm that works in a stationary and mobile scenario. It provides a route to nodes which forward data between each other directly in a single hop, or indirectly through multiple hops via neighbouring nodes. The algorithm works for static and mobile scenarios, whereby is very seemed to our proposal. However, their proposal is a heuristic that does not guarantee to obtain optimal values like our mathematical optimization model.

Likewise, in [13] the authors present a complete summary of algorithms in MWSN. However, any mathematical optimization model is proposed for a mobile wireless sensor network.

A summary of these related works is presented in Table 1. For understanding this table is necessary to consider the following observations:

- *Static* field refers to scenarios where all network nodes do not change its positions across time.
- *Mobile* field refers to scenarios where all network nodes can change its positions across time.
- *Delay-sensitive* field refers to scenarios where the delay is considered to be reduced as much as possible.
- *Optimal values* field refers to if was proposed a method to guarantee the obtaining of optimal values.
- *Prediction* field refers to if was proposed a method to predict future positions of network nodes.

According to Table 1, we can see that our proposal is the only one that accomplishes all fields.

On the other hand, remember that we are proposing two prediction algorithms: one based on Markov Chains and another based on Deep Learning. As follows, we are going to describe the related works for these two types of algorithms. In addition, it is necessary to take into account that our prediction algorithms use RSSI levels to know the distance between nodes instead of knowing accurate node positions through GPS devices with the aim of reducing energy consumption in the network and, thus, extending the network lifetime.

2) MARKOV CHAINS RELATED WORKS

According to our prediction algorithm based on Markov Chains, notice that it operates based on the distance of neighbour nodes to determine the best forwarding node in terms of delay and energy consumption requirements. That is, our Markov Chains design does not consider collecting the exact position (through GPS modules or a high amount of RSSI measurements) of neighbour nodes because, otherwise, it will require too much energy wasted for determining the exact position of each neighbour node, causing a high negative impact in terms of energy consumption in the network. Obviously, the optimal solution would be to know the exact position of neighbour nodes because it will allow us to know with high certainty the movement pattern of each neighbour node and, then, it would be the ideal method to select the best forwarding node. However, knowing the exact position of a node through RSSI measurements requires sending many control packets, which represents a high energy consumption in wireless sensor networks. In other words, our prediction algorithm based on Markov Chains only uses a few control packets (with RSSI information) to have a sense of distance between two pairs of nodes instead of using a high amount of control packets (with RSSI information) or GPS modules to obtain accurate nodes positions. In conclusion, with the aim of minimizing the energy consumption in the network, our prediction algorithm based on Markov Chains only uses distances without knowing the exact positions of nodes to determine the best forwarding node. For this reason, our Markov Chains method only considers knowing the distance, and, in this sense, it cannot be compared against Markov Chains methods that consider the exact position of neighbour nodes because they would be not comparable methods. Next, we are going to describe several related works based on Markov Chains.

In [23], the authors propose a multiuser multivariate multi-order Markov model and a multimodal user mobility pattern prediction approach. They propose a proposed to perform precise mobility pattern prediction based on real-world GPS trajectory data set. In other words, this method is based on GPS positions for predicting trajectories and, in this sense, it does not correspond to a method to be compared with our work due to the reasons previously explained.

In [24], the authors propose to integrate an attention technique into the Markov model to predict future locations. However, they predict locations based on a GPS dataset that has user coordinates. Similar to [23], this method is also based on GPS positions and, in this sense, it does not correspond to a method to be compared with our work due to the reasons previously explained.

In [25], the authors propose a weighted Markov prediction model based on mobile user classification. The trajectory information of a user is extracted first by analyzing real mobile communication data, where the complexity of a user's trajectory is measured using the location given by cellular base stations. That is, accurate positions are given by cellular networks, a technology that is not assumed and not common to be present in our wireless sensor network scenario. In this sense, this work does not correspond to a method to be compared with our work.

3) DEEP LEARNING RELATED WORKS

According to our prediction algorithm based on Deep Learning, it assumes the same considerations described for the

TABLE 1. Mathematical optimization related works summary.

Work	Static	Mobile	Delay Sensitive	Optimal values	Prediction
[11]	X	Only the sink			
[12]	X				
[13]	X	X	X		
[16]	X	Only the sink			
[17]	X	Only the sink			
[18]	X		X		
[19]	X	X	X		X
[20]	X	X			
Our proposal	X	X	X	X	X

prediction algorithm based on Markov Chains, that is, it operates based on the distance of neighbour nodes to determine the best forwarding node in terms of delay and energy consumption. Obviously, the selection of the best forwarding node is performed through a prediction model based on a deep learning technique instead of using a Markov Chains method. Likewise, our deep learning approach does not consider the exact position of neighbour nodes due to the reasons described previously for the prediction algorithm based on Markov Chains. In other words, our prediction algorithms, the Markov Chains approach and the Deep Learning approach, take into account the same considerations in order to be comparable, that is, both of them are based on distance instead of knowing the exact nodes' positions. Next, we are going to describe several related works based on Deep Learning techniques to predict the distance between nodes in order to select the best forwarding node.

In [26], the authors propose ML techniques to learn the mobility of the mobile mmWave (Millimeter-wave communication) users and predict their moving directions. The authors propose to use advanced MIMO antenna systems in cellular base stations to know exact user positions. Thus, based on this accurate information, they propose an ML approach to predict user mobility. The authors propose to use advanced MIMO base station techniques to know exact user positions. Thus, based on this accurate information, they propose an ML approach to predict user mobility. We cannot use this ML approach to be compared against our prediction algorithm based on Markov Chains due to two reasons: firstly, they assume knowing exact node positions, which is not assumed by our Markov Chains approach and, secondly, they use advanced antenna techniques appropriated to be employed in cellular networks, which is quite difficult to achieve in wireless sensor networks.

In [27], the authors propose a neural network framework to analyze citywide human mobility based on raw GPS data. The learning model is obtained considering exact user positions. For this reason, we cannot use this ML approach against our prediction algorithm based on Markov Chains due to they are not comparable.

In [28], the authors propose a deep learning approach to predict the exact future location of a node based on RSSI measurements in a dynamic environment. They use many RSSI control packets to determine the exact position for just one node. We cannot use this ML approach to be compared against our prediction algorithm based on Markov Chains due to two reasons: firstly, they assume to determine exact node positions, which is not assumed by our Markov Chains approach and, secondly, the fact of using many RSSI control packets to determine exact node positions would cause an excessive energy consumption in our wireless sensor network.

In [29], the authors propose a deep learning approach to determine the exact location of a device through RSSI measurements and interference detection. Similar to [28], many RSSI control packets are used to establish the exact device position. We cannot use this ML approach to be compared against our prediction algorithm based on Markov Chains due to two reasons: firstly, they assume to determine the exact device position, which is not assumed by our Markov Chains approach and, secondly, the fact of using many RSSI control packets would cause an excessive energy consumption in our wireless sensor network.

Finally, it was not possible to find works that build machine learning models based only on node distances instead of exact node positions. The methods used by these works require a high energy consumption or using technologies (advanced MIMO antenna systems) that cannot be allowed in wireless sensor networks. For this reason, in this work, we also propose a deep learning approach based on node distances in order to be compared against our Markov Chains approach under the same considerations.

C. CONTRIBUTIONS

- We propose a mathematical optimization model for MWSN considering that absolutely all network nodes are mobile, while many works present a mixed scenario where some nodes are static and some are mobile, but not all network nodes are mobile. However, if a work assumes all network nodes are mobile, it proposes an algorithm instead of proposing a mathematical optimization model. Given these previous reasons, we consider our proposal is novel in MWSN.
- The results obtained by the mathematical optimization model could serve as a reference to evaluate new algorithms in order to analyze their performance in terms

of delay and energy consumption in MWSN. In other words, optimal values given by the mathematical model can be used to determine how good are the results of a new algorithm proposed for MWSN applications considering, obviously, the same goals and parameters of the mathematical model.

- We proposed an algorithm based on Markov Chains to predict if neighboring nodes will be far or close in order to avoid future interruptions in communication paths. This strategy allowed us to establish more reliable communication paths, which reduced considerably the end-to-end delay in scenarios where the number of nodes was scarce. In addition, in order to contrast the Markov Chains results, we have proposed a Deep Learning approach to select the best forwarding node taking into account only distances between nodes instead of using the exact node positions. This feature makes our approach very special due to most works consider exact node positions to predict node mobility. In summary, our approaches, with only a sense of distance between nodes, are able to predict node mobility minimizing at the same time the delay and energy consumption in the network.
- While many algorithms are based on accurate positions given by GPS devices, our prediction algorithms use RSSI levels to avoid using GPS devices with the aim of reducing energy consumption in the network and, thus, extending the network lifetime.

III. PROPOSAL

A. MATHEMATICAL MODEL FORMULATION

In this section, we propose a multi-objective mathematical optimization model to build a path from a source node to the sink minimizing the delay and energy consumption of the network. The mathematical model needs too much time to provide a solution, whereby it is not an affordable and scalable solution for real mobile wireless sensor networks applications because they require solutions that must be obtained as fast as possible. However, in despite of wasting a considerable time to provide a solution, our mathematical optimization model is used as an offline method that has global information about the network, which allows us to obtain the best possible solution, that is, the optimal solution value for a specific network scenario. For this reason, it is obvious that the mathematical model will always obtain the best results for all metrics evaluated later in IV. As a consequence, the optimal solutions offered by the mathematical model can be used as reference values to evaluate how close is the performance of the algorithms proposed later.

1) PROBLEM DESCRIPTION AND ASSUMPTIONS

In this section, our problem is enunciated and described in detail, as well as some assumptions are shown in order to simplify our mathematical optimization model.

Based on Figure 2, we will describe our problem:

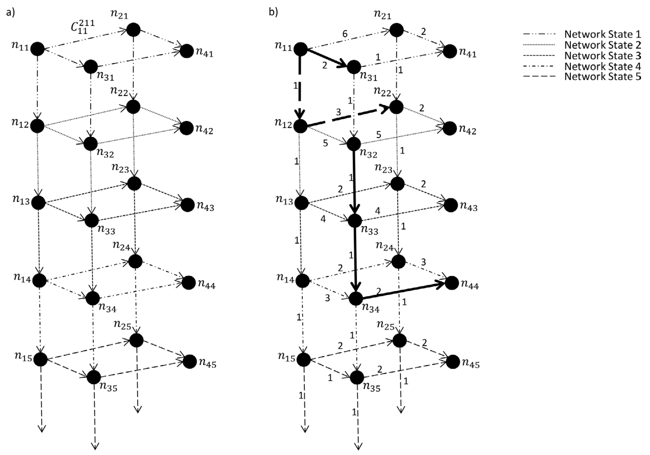
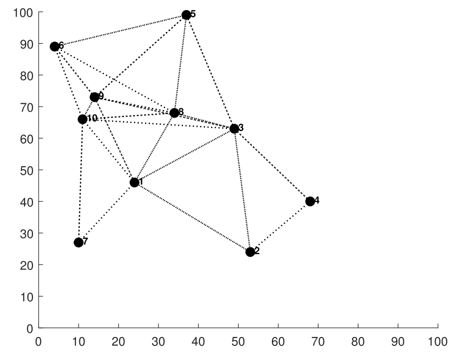
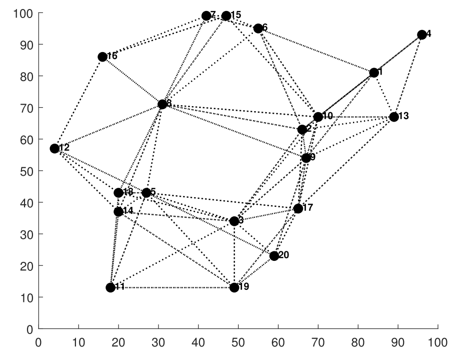


FIGURE 2. Problem scenario.

- **Mobile Network:** Assume we have a mobile network at which network nodes' position changes across time periods. For this reason, the links cost between network nodes also changes across time periods. This means that at each time period the network has particular links costs. For this reason, we could say these particular links costs reflect the network state at a given time period. In this sense, each network at a given time period will be called *Network State*. For instance, *Network State* at time period *1* is called *Network State 1*, *Network State* at time period *2* is called *Network State 2*, and so on. In other words, according to Figure 2.a we have an initial network (Network State 1) compound by four nodes. Due to these nodes conform a network, there are interrelations between them that we will call *Links*. These links have a cost, which can be represented, for example, by the distance, the delay, or the energy consumption. In this work, we consider two types of cost: delay cost and energy consumption cost. In the next time period, network costs at the Network State 1 change, and then, these new interrelations between the nodes are now the *Network State 2*. As the next time period occurs, the network at Network State 2 becomes the network at the Network State 3, and this network will be the network at the network State 4, and so on. For example, Figure 3.a shows a mobile network of 10 nodes at a given time, that is, at a network state i. Due to the network will move in the next time frame, the link costs of the network will change, generating a new network state $i+1$. As the network moves for each time frame, a new network state is generated to represent the network movement across time, as we can generically see in Figure 2. The same logic applies to a bigger scenario, for example, a network of 20 nodes in Figure 3.b.
- **Nodes:** Each node is denoted as n_{it} where i is the number node and t is the network state of the node. Depending on the communication range, a node can communicate with another node in the direction described by the Figure 2. For example, n_{11} can communicate with n_{21} and n_{31}.



(a) Network of 10 nodes



(b) Network of 20 nodes

FIGURE 3. Typical networks.

TABLE 2. Sets, parameters and variables for the mathematical model.

Sets	Description
N	Set of network nodes.
S	Set of network states.
T	Set of destination nodes.
Parameters	**Description**
Del_{it}^{jul}	Delay cost at network state l required to go from node i at state t to node j at state u.
Ec_{it}^{jul}	Energy consumption cost at network state l required to go from node i at state t to node j at state u.
Variables	**Description**
X_{it}^{juld}	Determines if link (i, j) at state l that goes from node i at state t to node j at state u is selected for building a path to reach destination node d (Binary variable).
Y_{il}^{d}	Determines if node i at state l is selected as a forwarding node for building a path to reach destination node d (Binary variable).
D_{jl}^{d}	Defines if node j is selected at destination state l for destination node d (Binary variable).
DS_{l}^{d}	Determines if network state l is selected as destination state for the destination node d (Binary variable).

- **Buffers:** In telecommunication networks, a router or a sensor (a node) can decide not-sending its message, storing it in a buffer until will be appropriate to send it to another node. In our model, this situation is represented as a link between n_{11} and n_{12}, meaning that n_{11} can store its message in its buffer, that is, node n_{12}.
- **Costs:** As was mentioned before, a link has a cost. Then, there is a cost for sending a message from n_{11} to n_{21} called C_{11}^{21l}, and denoted as C_{it}^{jul}. This expression is the cost to carry a message from node i at the state t to node j at the state u at the Network State l. As we mentioned previously, we have two types of costs: delay cost and energy consumption cost. The cost C_{it}^{jul} is a general form (for illustrative reasons) for representing any cost in our network.
- **Directed graph:** In Figure 2, our goal consists to carry a message from node 1 to nodes 2 and 4. Then, our *Source* node is the node 1, and our *Destination* nodes are the nodes 2 and 4. In this sense, a directed graph is constructed from Source node to Destination nodes. For this reason, links direction points to the Destination nodes.
- **Goal:** Our goal consists of carrying a message from a Source node to a Destination node using neighboring nodes as forwarding nodes for sending a message, and even using buffers, if it is necessary, for waiting an appropriate situation for sending the message. In this sense, we have to find the minimum cost path between a Source node and a Destination node considering the network is changing across time, which is represented by Network States. Additionally, for simplicity, we assume only one link can be selected for sending the message per each Network State. This means that if a message is at the node n_{11}, this node at this Network State 1 can send a message to only one neighbour, n_{21} or n_{11}, or storing it in its buffer, that is, n_{12}.
- **Example Result:** According to the example shown in Figure 2.b and based on links cost, the minimum cost path from the Source node n_{11} to the Destination node 4 is the path compounded by the highlighted links: n_{11} to n_{31}, n_{32} to n_{33}, n_{33} to n_{34} and n_{34} to n_{44}. In other words, $X_{11}^{3114} = 1$, $X_{32}^{3324} = 1$, $X_{33}^{3434} = 1$ and $X_{34}^{4444} = 1$. Likewise, the minimum cost path from the Source node n_{11} to the Destination node 2, is the path compounded by the highlighted links: n_{11} to n_{12} and n_{12} to n_{22}. In other words, $X_{11}^{1212} = 1$ and $X_{12}^{2222} = 1$. Notice that the solution does not correspond to the path n_{11} to n_{21} due to its cost, 6, is higher than the optimal solution provided by our proposal, that is, 4.

2) SETS, PARAMETERS AND VARIABLES

Next, the sets, parameters, and decision variables of the mathematical optimization model are summarized in Table 2.

3) OBJECTIVE FUNCTIONS

Next, the mathematical optimization model is described as follows:

$$min \sum_{i\in N}\sum_{t\in S}\sum_{j\in N}\sum_{u\in S}\sum_{l\in S}\sum_{d\in T}(w_1 * F_1 + w_2 * F_2)$$
$$where\ F_1 = Del_{it}^{jul} * X_{it}^{juld},\ F_2 = Ec_{it}^{jul} * X_{it}^{juld}\ and$$
$$w_1 + w_2 = 1 \tag{1}$$

This expression indicates we are going to obtain a solution that combines delay and energy consumption requirements. Otherwise, if we consider only one function, for example, the delay function, the solution will only minimize the delay without taking into account the energy consumption, which is very disadvantageous in the context of wireless sensor networks because they have limited energy levels that we have to preserve as much as possible. Likewise, if we only consider the energy consumption function, the solution will not take into account the appearance of long delays, which will cause a negative impact on delay-sensitive wireless sensor networks.

4) CONSTRAINTS

Subject to:

$$\sum_{l\in S|l>1} D_{jl}^d = 1 \quad \forall j\in N \quad \forall d\in T \tag{2}$$
$$\sum_{l\in S} D_{jl}^d = 0 \quad \forall j\in N \mid j\neq d \quad \forall d\in T \tag{3}$$
$$D_{jl}^d * DS_l^d = D_{jl}^d \quad \forall j\in N \quad \forall l\in S \quad \forall d\in T \tag{4}$$
$$\sum_{l\in S} DS_l^d = 1 \quad \forall d\in T \tag{5}$$
$$DS_l^d = 0 \quad \forall l\in S \mid l=1 \quad \forall d\in T \tag{6}$$
$$DS_l^d \sum_{i\in N}\sum_{t\in S}\sum_{j\in N}\sum_{u\in S} X_{it}^{juld} Y_{im}^d D_{jl}^d = DS_l^d$$
$$\forall l,m\in S \mid m=l-1 \quad \forall d\in T \tag{7}$$
$$DS_l^d \sum_{i\in N} Y_{im}^d = DS_l^d \quad \forall l,m\in S \mid m\leq l \quad \forall d\in T \tag{8}$$
$$DS_l^d \sum_{i\in N} Y_{im}^d = 0 \quad \forall l,m\in S \mid m>l \quad \forall d\in T \tag{9}$$
$$\sum_{i\in N}\sum_{t\in S}\sum_{u\in S}\sum_{l\in S} X_{it}^{juld} = 1 \quad \forall j\in N \mid j=d \quad \forall d\in T \tag{10}$$
$$DS_l^d \sum_{i\in N}\sum_{t\in S}\sum_{j\in N}\sum_{u\in S} X_{it}^{jumd} Y_{in}^d Y_{jm}^d = DS_l^d$$
$$\forall l,m,n\in S \mid m>1 \wedge m=l \wedge n=m-1 \quad \forall d\in T \tag{11}$$
$$DS_l^d \sum_{i\in N}\sum_{t\in S}\sum_{j\in N}\sum_{u\in S} X_{it}^{jumd} = DS_l^d$$
$$\forall l,m\in S \mid m\leq l \quad \forall d\in T \tag{12}$$
$$\sum_{i\in N}\sum_{t\in S}\sum_{j\in N}\sum_{u\in S} X_{it}^{juld} Y_{jl}^d = 1$$
$$\forall i\in N \mid i=Source \quad \forall l\in S \mid l=1 \quad \forall d\in T \tag{13}$$

Equation 1 corresponds to the general objective function, which is composed of two objective functions: one based on the delay cost and the other on the energy consumption cost. The previous expressions are explained in the following items:

- Destination State Constraints (from 2 to 10): The following expressions are referred to the Destination State; that is, the network state at which a Destination node is found at the minimum possible cost.
  - Defining D_{jl}^d: D_{jl}^d allows to obtain the Destination State l at which a Destination node j is found at the minimum possible cost. The expression 2 avoids that D_{jl}^d will be one at first state. The equation 3 avoids D_{jl}^d will be one for nodes different from the destination node d.
  - Defining DS_l^d: DS_l^d allows to determine the Destination State l at which a Destination node d has been found at minimal cost. Expression 4 allows us to know the state l at which D_{jl}^d was selected. Equation 5 indicates that only one destination state is possible. In the expression 6 we assume it is not possible that the destination state will be the first state.
  - Selecting forwarding nodes: A forwarding node indicates the node selected at each state for building the minimum cost path. Expressions 7 and 8 restrict to one the number of Y_{jl}^d for each State that is less than the Destination State. The equation 9 restricts to zero the number of Y_{jl}^d for each State that is higher than the Destination State. The expression 10 indicates that it is possible only one link to the Destination node for all states; that is, only one state is selected, and for the rest of the states, the link must be zero.
- Source State Constraint: The Source State indicates the network state at which the Source node starts to build the minimum cost path. The expression 11 restricts to one the number of X_{it}^{juld} for the Source State.
- Intermediate State Constraints: These constraints allow us to select the predecessor node Y_{im}^d based on the current forwarding node Y_{jl}^d. In order to understand what these two types of nodes mean, let's see an example. If there is a directed link that goes from 1 to 2, the current forwarding node is 2, and the predecessor node is 1. The expression 11 allows us to select predecessor nodes at intermediate states, where intermediate states refer to the network states involved between a Destination State and a Source State. The equation 12 restricts to one the number of X_{it}^{juld} for each network state that is equal or less than the Destination State.
- Defining the First State Solution Constraint: All constraints described above allow finding the minimum cost path between a Source node and a Destination node d. However, up to now, our model does not consider that the Destination State could be at the first network state; that

Algorithm 1 Post-Processing Pseudocode.

```
1: parameters $Source, Destination\ d$
2: $minSolution = MathModel(Source, Destination\ d)$
3: $costFirstState = C_{it}^{jul} \quad | \quad i = Source, j = Destination, t = u = l = 1$
4: if $costFirstState < minSolution$ then
5:     $minSolution = costFirstState$
6: end if
```

is, the minimum cost path to reach a destination node d could be at the first network state. For this reason, it is necessary to apply the following post-processing pseudocode:

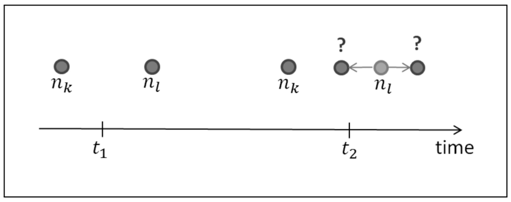
This pseudocode basically indicates that if there is a directed link between a source node and a destination node d at which its cost is less than the solution found by the mathematical optimization model. In such case, the optimal solution is at the first network state. Otherwise, the solution is in a future network state calculated by the mathematical optimization model.

Notice that our multi-objective mathematical optimization model was designed for finding paths from a single source node to multiple sinks (destination nodes). However, later, in the Results Section (IV), we will see that the network scenario simulation only considers one source node and one sink. In this sense, our mathematical model is configured to find a path between one source node and one sink.

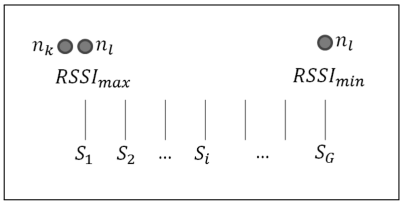
B. MOBILITY PREDICTION ALGORITHM BASED ON MARKOV CHAINS

In this section, we propose the use of a mobility prediction algorithm based on Markov Chains to estimate future distances between nodes in a mobile network. Forecasting future distances help us to determine if future movements of nodes will cause communication disruptions in paths. Managing this information provided by the mobility prediction algorithm can be useful for decreasing communication disruptions, and therefore, result in the reduction of the end-to-end delay in the mobile network. The details about why the mobility prediction algorithm can reduce the end-to-end delay of the network were described in Section II-A, specifically, in Figures 1.a and 1.b.

In order to be aware of network mobility, we use RSSI (Received Signal Strength Indicator) measurements, which indicate an approximated distance measurement between a pair of nodes. Specifically, our mobility prediction method allows each network node to estimate future distances (RSSI measurement) of neighboring nodes for determining if they will be farther or closer at a future time. In summary, we propose to use a mobility prediction method based on Markov Chains for estimating future RSSI measurements, which will help us to determine if a node will cause a communication disruption at a future time. Managing this information will be useful to minimize the delay experimented in the network. Details about how we manage this information are explained as follows:



(a) Possible movement of n_l.



(b) RSSI States.

FIGURE 4. Defining Markov states.

In relation to the Figure 4.a), suppose we have a network that consists of two nodes: n_k and n_l, where n_l is a neighboring node of n_k. There are two times, t_1 and t_2, at which our small network is evolving in time. At time t_1 the node n_l is located at a certain distance from n_k. However, at time t_2 we want to predict if n_l will be farther or closer (or at the same distance in t_1) from n_k.

Additionally, according to Figure 4.b), there is a minimum and maximum distance at which n_l can be located to establish a communication link with n_k. At a minimum distance, n_l will have a maximum RSSI, $RSSI_{max}$, and, at a maximum distance, n_l will have a minimum RSSI, $RSSI_{min}$. At t_2, n_l could be located at any distance between $RSSI_{min}$ and $RSSI_{max}$. Our goal consists of estimating a location between $RSSI_{min}$ and $RSSI_{max}$ at which n_l will be in a future time (in this case, t_2). Theoretically, there are infinite locations between $RSSI_{min}$ and $RSSI_{max}$, but for our model, we assume discrete locations equitably distributed. These possible locations, at which n_l could be, we call them *states*. In this sense, at a future time t_2, n_l could be at S_1, S_2, S_r or S_G, where G is the maximum number of states. The initial probability of n_l for being at any state S_i is $1/G$, which is called *Initial Probability Distribution of set S* (π), which can be expressed as follows:

$$\pi = \{P_{s_1}, P_{s_2}, \ldots, P_{s_G}\} \tag{14}$$

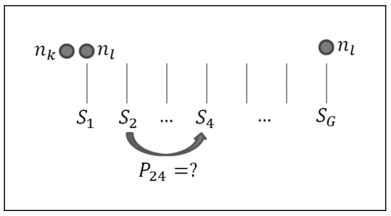
According to Figure 5.a), suppose we want to know the probability to go from the state S_2 to the state S_4, which is calculated with the following expression:

$$P_{24} = \frac{N(S_2, S_4)}{\sum_{j=1}^{G} N(S_2, S_j)} \tag{15}$$

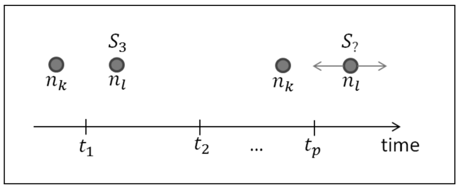
where $N(S_i, S_j)$ is the number of times that state S_i follows state S_i.

This expression can be applied for the rest of the probabilities using the following expression:

$$P_{ij} = \frac{N(S_i, S_j)}{\sum_{j=1}^{G} N(S_i, S_j)} \tag{16}$$



(a) Probability to go from state S_2 to S_4



(b) Predicting the future state of n_l

FIGURE 5. Defining Markov states.

In this sense, we have the probability to go from any state S_i to any state S_j. These probabilities can be expressed in a matrix, which is called *Transition Matrix*:

$$T = \begin{bmatrix} P_{11} & P_{12} & \dots & P_{1G} \\ P_{21} & P_{22} & \dots & P_{2G} \\ . & . & \dots & . \\ . & . & \dots & . \\ P_{G1} & P_{G2} & \dots & P_{GG} \end{bmatrix} \tag{17}$$

In relation to the Figure 5.b), suppose that in a current time t_1, n_l is at state S_3 and we want to estimate the future state of n_l at a future time t_p. For this purpose, we can apply the following expressions:

$$\pi_p = \pi * T^p \tag{18}$$

$$S_p = max\{\pi_p\} \tag{19}$$

$$S_p = max\{P_{s_1}, P_{s_2}, \dots, P_{s_G}\} \tag{20}$$

According to the expression 20, n_k can finally obtain the most probable future state at which n_l will be at time t_p, and we can use this information for routing decisions in order to reduce the delay caused for communication disruptions in paths. More precisely, these routing decisions are made by our routing algorithm which, in addition to our mobility prediction algorithm, allows us to reduce the end-to-end delay and minimize the energy consumption of network nodes.

Based on the previous stochastic model proposed, we present our mobility prediction algorithm. The pseudocode of this algorithm is shown as follows.

This algorithm pretends to find the forwarding node with the best probability to be near to the current node in order to find as fast as possible the paths between a source node and multiple sinks. The Transition Matrix is calculated for each node at each period time t in order to update this matrix with the aim of select the most appropriated forwarding node in terms of delay and energy consumption. Lines 1 to 3 initialize the sensor nodes and the sink. Line 4 indicates that for each time period, it is required to check where is a data packet in

Algorithm 2 Prediction Algorithm Based on Markov Chains Pseudocode.

```
1:  Initialize Path = []
2:  Initialize Sensor Nodes = [$N_1 N_2 \dots N_n$]
3:  Initialize Sink = [$S_1$]
4:  for $t = 1$ to totalTimePeriods do
5:      if A packet arrived to a $N_i$ then
6:          Obtain list of neighbours V
7:          Obtain score of each $V_j$ based on Probability Transition
8:          Matrix and energy level
9:          if $|V| \neq 0$ then
10:             if $V_j$ is the sink $S_1$ then
11:                 Send message to $S_1$
12:                 Add $S_1$ to Path
13:             end if
14:             if $V_j$ is a connected node then
15:                 Send message to the connected node $V_j$
16:             end if
17:             if $V_j$ is not a connected node nor the sink then
18:                 Send message to the best $V_j$ based on its score
19:             end if
20:         else
21:             Store the message in the buffer until the next time
22:             period t
23:         end if
24:     end if
25: end for
```

order to be sent through several forwarding nodes to finally achieve the sink (line 5). If the data packet is at node i, then we obtain the list of neighbour V (line 6). From that list, the best forwarding node is obtained through the Probability Transition Matrix (lines 7 and 8). If the node i does not have neighbour nodes, the node i stores the data packet until the next time period (lines 9, 20-23). If between the neighbour nodes there is the sink, we send the data packet to the sink, and then, the path has been built (lines 10 to 13). If between the neighbour nodes there is a *connected node*, we send the data packet to it because it has high chances to find faster the sink (lines 14 to 16). Finally, if between the neighbour nodes there is not a connected node nor the sink, we send the data packet to the best forwarding node obtained in the lines 7 and 8. For the next time period, this process is repeated until the sink is achieved. Once the sink is achieved, the path between a source node and the sink has been built and the algorithm finishes. In terms of computational complexity, we use the *Big-O* notation to indicate the time complexity of our Markov Chains approach. In this sense and according to algorithm 2, our prediction algorithm based on Markov Chains is $O(T * V * G)$; where: T is the total number of time periods (line 4), V is the total number of neighbour nodes of a node N_i (line 6) and, G corresponds to the number of discrete states for calculating the Transition Matrix (line 7).

C. MOBILITY PREDICTION ALGORITHM BASED ON DEEP LEARNING

In this section, we also propose a deep learning approach to predict future distances between nodes in a mobile network. Predicting future distance helps us to determine if future movements of nodes will cause communication disruptions in paths. For this reason, handling this information provided by the mobility prediction algorithm can be useful for decreasing communication disruptions, and therefore, result in the

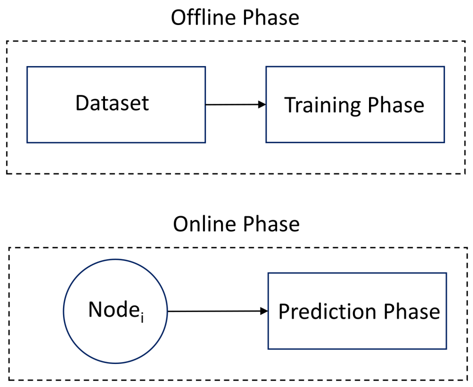


FIGURE 6. **Deep learning approach.**

reduction of the end-to-end delay in the mobile network. The details of this approach are described in the following items.

1) DEEP LEARNING APPROACH

We propose to use a supervised deep multi-layer perceptron (DMLP) neural network for predicting future nodes' positions in a mobile network taking into account the same considerations seen with the Markov Chains approach. These considerations are described as follows:

- In order to be aware of network mobility, we use RSSI (Received Signal Strength Indicator) measurements, which indicate an approximated distance measurement between a pair of nodes. Specifically, our deep learning method allows each network node to estimate future distance measurements (RSSI levels) of neighboring nodes for determining if they will be farther or closer at a future time. This will help us to determine if a node will cause a communication disruption at a future time. Handling this information will be useful to minimize the delay and the energy consumption experimented in the network.
- With the aim of minimizing the delay and the energy consumption in the network, this deep learning approach pretends to find the forwarding node with the best chances to be near to the current node (the node that has a packet and has to send it to a neighbour node) in order to build as fast as possible a path between a source node and a sink.

Taking into account the previous considerations, Figure 6 shows a diagram that summarizes the operation of our deep learning approach.

This diagram (Figure 6) is divided in two phases: *Offline Phase* and *Online Phase*, which are described in detail as follows:

- *Offline Phase:* The offline phase is used to build a deep learning model to classify the best forwarding node to be selected in terms of minimum distance and energy consumption. A dataset compound of many samples is used to feed a training phase in order to obtain a learning model for a classification task. The dataset and the classification proposal are explained in detail in the *Dataset* section. This offline phase is performed outside of the network simulator operation described in III-D.
- *Online Phase:* The online phase predicts in real time the best forwarding node to be selected in terms of minimum distance and energy consumption. In detail, each node i has an intern prediction phase to determine the best forwarding node when it will be required, that is, when a node i has a packet that has to be sent to a forwarding node in order to build a path to achieve the sink.

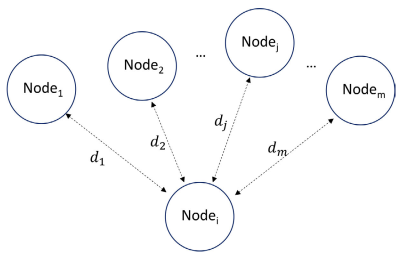


FIGURE 7. **Dataset for node i.**

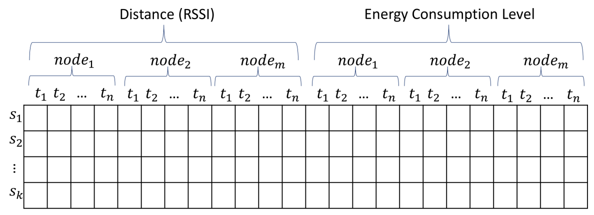


FIGURE 8. **Dataset features.**

2) DATASET

A dataset compound of many samples is used to feed a training phase in order to classify the best forwarding node to be selected in terms of minimum distance and energy consumption. This dataset represents the fact of having several neighbour nodes with a certain distance (RSSI level) from a node i and a certain energy consumption level. In other words, each neighbour node has a distance in relation to a node i and also has an energy consumption level. These values, distance, and energy consumption, must be denoted across time in order to predict the best forwarding node in the future. In detail, each neighbour node j must describe its distance (respect to a node i that has a message packet) and energy consumption level at a different time in order to know which neighbour node will be farther or closer in the future (See Figure 7). The samples of this dataset are shown in Figure 8.

This dataset is described in detail as follows:

- s_1 to s_k correspond to all samples from which the deep learning model will learn. Notice that this learning process corresponds to an offline phase.
- Each s_i has two types of values: distance values and energy consumption values.
- At each s_k, distance values are assigned for each neighbour node j, in this case, node 1, node 2 and node m.

These values are manually assigned in order to fill out all of these values for the training phase. In addition, each neighbour node j has a distance value related to a specific time t_p in order to represent the fact that this neighbour node is moving, and then, this movement means that distance changes across time. Each neighbour node j has, from t_1 to t_n, to represent its movement in terms of distance values.

- Similar to distance values, at each s_k, energy consumption values are also assigned for each neighbour node j. These values are also manually assigned in order to fill out all of these values for the training phase. In addition, each neighbour node j has an energy consumption value related to a specific time t_p in order to represent the fact that this value can change across time. Each neighbour node j may have, from t_1 to t_n, different energy consumption values.
- For each s_k, the best forwarding node j in terms of distance and energy consumption is labeled. In other words, from node 1 to m, one of them is selected to be the best forwarding node. In this sense, this case corresponds to a multiclass classification problem, at which from many categories (neighbour nodes), one of them must be labeled as the correct one. Once each sample s_k is labeled, a training phase is launched in order to create a learning model according to this dataset.
- In a previous item, we said that each neighbour node i had several neighbour nodes from node 1 to node m. Initially, the default value of m was assumed to be equal to $N - 1$, where N was the total number of network nodes, and minus 1 to omit the node i. However, after several tests taking into account a network of maximum of 50 nodes, a communication radius of 20m and the Markov-Gauss Mobility model, we obtained that, on average, the maximum number of neighbour nodes for a node i was 8. In this sense, there was a significant reduction of the number of features, going from $N-1$ to only 8.
- In a previous item, we said that each neighbour node j had a distance and an energy consumption value for each time from t_1 to t_n. The n value was considered for three scenarios: 5, 10, and 15. $n = 5$ means that it will be considered the last five values of distance and energy consumption to determine the best forwarding node. Likewise, for $n = 10$ and $n = 15$ it will be considered the last ten and fifteen values, respectively, of distance and energy consumption to determine the best forwarding node. It is necessary to test several values of n in order to determine the best results in terms of delay and energy consumption in the network simulator, which is described later.
- Considering the previous items, and taking into account that $m = 8$, n can be at least 15, and there are distance and energy consumption values, each sample s_k has 240 columns (8×15x2). As we said previously, an additional column is added to each s_k in order to label it, indicating the best forwarding node j in terms of distance and energy consumption.

TABLE 3. Parameters and results for training and testing phases.

Parameters	For $n = 5$	For $n = 10$	For $n = 15$
Training size	3000 samples	3000 samples	3000 samples
Testing size	2000 samples	2000 samples	2000 samples
Training time (sec)	2575	3077	3315
Testing time (sec)	232	274	295
TP	24,57%	22,36%	20,09%
TN	72,59%	75,44%	78,33%
FP	1,73%	0,94%	0,84 %
FN	1,11%	1,26%	0,74%
Accuracy	0,9716	0,978	0,9842
Precision	0,9342	0,9596	0,9598
Recall	0,9567	0,9466	0,9644
F1	0,9453	0,9531	0,9621

Once each sample s_k is labeled, a training phase is launched in order to create a learning model according to this dataset. The dataset was divided into two groups: one group for the training phase, and another group for the prediction phase (testing phase). Table 3 summarizes the parameters and results for the training and testing phases. For this classification problem, we used the Deep Learning Toolbox in MATLAB. In detail, we configured a five-layer neuronal network with a *sofmax* activation function. For each sample s_k, and once our model has been trained, this softmax activation function shows us the probability of each neighbour node to be the best forwarding node. In other words, if a node i has eight neighbour nodes with certain values of distance and energy consumption across time, then, the softmax activation function shows a list $\{P_1, P_2, \ldots P_8\}$ where P_i is the probability of the neighbour node i to be the best forwarding node and, $\sum_{i=1}^{8} P_i = 1$. In this sense, the best forwarding node corresponds to the neighbour node with highest probability, that is, $P_{best} = max\{P_1, P_2, \ldots P_8\}$. The training and testing results are shown in Table 3.

From Table 3 is necessary to remark the following details:

- The deep learning model obtained through the training phase is applied later in the network simulator to select the best forwarding node for a node i when it has a packet that needs to be sent to any of its neighbour nodes j.
- The higher is n, better values are obtained for accuracy, precision, recall, and F1. In other words, a higher value of n allows the learning model to have more distance information in order to improve the distance prediction of its neighbours.

Based on the description of our Deep Learning approach, we present the pseudocode of how this approach is incorporated in our network simulator, which will be described later, to select the best forwarding neighbour node in terms of delay and energy consumption. The pseudocode of this algorithm is shown the Algorithm 3.

This algorithm pretends to find the best forwarding node in terms of delay and energy consumption according to the explanation described in section III-C2. Lines 1 to 3 initialize the sensor nodes and the sink. Line 4 indicates that for each time period, it is required to check where is a data packet in

Algorithm 3 Prediction Algorithm Based on Deep Learning Pseudocode.

```
1:  Initialize Path = []
2:  Initialize Sensor Nodes = [$N_1 N_2 \dots N_n$]
3:  Initialize Sink = [$S_1$]
4:  for $t = 1$ to totalTimePeriods do
5:      if A packet arrived to a $N_i$ then
6:          Obtain list of neighbours $V$
7:          Obtain the best forwarding node $V_j$ applying the Deep
8:          Learning Prediction phase
9:          if $|V| \neq 0$ then
10:             if $V_j$ is the sink $S_1$ then
11:                 Send message to $S_1$
12:                 Add $S_1$ to Path
13:             end if
14:             if $V_j$ is a connected node then
15:                 Send message to the connected node $V_j$
16:             end if
17:             if $V_j$ is not a connected node nor the sink then
18:                 Send message to the best $V_j$ based on its score
19:             end if
20:         else
21:             Store the message in the buffer until the next time
22:             period $t$
23:         end if
24:     end if
25: end for
```

order to be sent through several forwarding nodes to finally achieve the sink (line 5). If the data packet is at node i, then we obtain the list of neighbour V (line 6). From that list, the best forwarding node is obtained through the Prediction Phase of the Deep Learning approach (lines 7 and 8). If the node i does not have neighbour nodes, the node i stores the data packet until the next time period (lines 9, 20-23). If between the neighbour nodes there is the sink, we send the data packet to the sink, and then, the path has been built (lines 10 to 13). If between the neighbour nodes there is a *connected node*, we send the data packet to it because it has high chances to find faster the sink (lines 14 to 16). Finally, if between the neighbour nodes there is not a connected node nor the sink, we send the data packet to the best forwarding node obtained in lines 7 and 8. For the next time period, this process is repeated until the sink is achieved. Once the sink is achieved, the path between a source node and the sink has been built and the algorithm finishes. In terms of computational complexity, we use the *Big-O* notation to indicate the time complexity of our Deep Learning approach. In this sense and according to algorithm 3, our prediction algorithm based on Deep Learning is $O(T * V)$; where: T is the total number of time periods (line 4) and V is the total number of neighbour nodes of a node N_i (line 7). The complexity obtained for the Deep Learning approach ($O(T * V)$) is less than the Markov Chains method ($O(T * V * G)$). For this reason, in terms of computational complexity, the Deep Learning approach is recommended to be used for our problem.

D. NETWORK SIMULATOR

In order to test our prediction distributed routing algorithm based on Markov Chains and the Deep Learning approach, we have designed a Mobile Wireless Sensor Network Simulator in MATLAB, which has the following basic network components:

- *Destination node:* It is the final node that will receive a data message. In our simulations, this node will always be the last network node.
- Source node: This node will have a data message, which must arrive to the destination node. In our simulations, this node will always be the first network node.
- *Connected node:* If a message arrives to this node, this node knows the path to achieve the destination node. This is a technique that helps us to find faster the sink when the data packet is close to it.
- *Forwarding node selection:* When a node has a data message, this process consists to select properly a neighbour node as a forwarding node, which is selected according to the following priorities: If among the neighbour nodes there is the destination node, then, the forwarding node is the destination node. If among the neighbour nodes there is not the destination node, but there is a connected node, then, the forwarding node is the connected node. If among the neighbour nodes there is not a destination node neither a connected node, then, the forwarding node is a node obtained by the Prediction method selected: the Markov Chains approach or the Deep Learning approach.
- *Sink refreshing:* This process consists to determine which nodes will be *connected nodes* at each certain period. This refreshing process is required due to network mobility, since it causes that connected nodes established in a previous state period, they will not possibly be *connected nodes* in the next period.
- *Loop detection:* It is a very important process in order to avoid that a packet remains in a loop.
- *Prediction at each k-state with the Markov Chains approach:* At each network state the Transition Matrix (T) is calculated for all network nodes, except the destination node. Remember that this Transition Matrix stores the probability of each node to be at certain distance level respect to their neighbour nodes.
- *Prediction at each k-state with the Deep Learning approach:* At each network state and for each node i, the last n distance, and energy consumption values are considered to determine the best forwarding node.
- *Prediction for selecting a forwarding node:* As we said before, if among the neighbour nodes there is not a destination node neither a connected node, then, the forwarding node is a node given by the Prediction method. In the case of the Markov Chains approach, this forwarding node is selected based on the information given by the Transition Matrix (see lines 7 and 8 in Algorithm 2). On the other hand, in the case of the Deep Learning approach, this forwarding node is selected based on the information given by the *Prediction Phase* (see lines 7 and 8 in Algorithm 3).

E. ENERGY MODEL

The energy consumption model is required to take into account special considerations. If a node has to send a data packet of K bits to another node located at a D distance, then, the following are the expressions to calculate the energy consumption in the transmitter node as well as the receiver node. In the transmitter node, the consumption is $E_{elec}+E_{amp}$, where E_{elec} is the energy consumption for codification, modulation and filtering. E_{amp} corresponds to energy consumption for the *Transmitter Power Amplifier*. In the same way, in the receiver node, the consumption corresponds to E_{amp}. Then, the expressions for the transmitter and receiver sensor are the following [19]:

$$E_{tx} = (E_{elec} + E_{amp}) * K * D^2 \tag{21}$$
$$E_{rx} = E_{amp} * K * D^2 \tag{22}$$

In the constraint 21 there will have a higher energy consumption than constraint 22 because for transmission is required an extra consumption for codification, modulation and filtering (E_{elec}), in addition to energy consumption for amplifying the signal received (E_{amp}). In detail, in the context of this work is possible to send two types of packets: data packets and control packets. A data packet corresponds to information that is collected by a source sensor node and, then, this source node needs to build a path in order to send this data packet to the sink. Thus, when a data packet has to be sent to a neighbour node, the expression 21 is applied to the node that sends the data packet and, the expression 22 is applied to the node that receives the data packet. In this sense, this data packet is sent several times through different network nodes until it achieves the sink. The data packet size corresponds to the value indicated in 4. On the other hand, in the context of our work, a control packet is used to collect information about the distance of neighbour nodes in order to build a path according to the indications given for our prediction algorithms, that is, the prediction algorithm based on Markov Chains and the prediction algorithm based on Deep Learning. The control packet size corresponds to the value indicated in 4.

F. MOBILITY MODEL

With respect to the network nodes movement, the present approach was evaluated considering a Gauss-Markov mobility model [9], [20], at which the mobility network was configured for not being totally random in order to be predictable because, otherwise, there would not have any reason of applying a prediction method in scenarios where the movements of the sensors are totally randomized. In other words, the mobility network must be predictable in a certain manner since we are dealing with sensors attached to objects, animals, or humans which exhibit movements that are not totally randomized, that is, present a certain movement pattern.

In this model, the values of the mobility speed and movement direction of the node at each instant are calculated only on the basis of those values in the previous instants as follows:

$$v_n = \alpha v_{n-1} + (1-\alpha)\overline{v} + \sqrt{(1-\alpha^2)}v_{x_n-1} \tag{23}$$
$$\theta_n = \alpha \theta_{n-1} + (1-\alpha)\overline{\theta} + \sqrt{(1-\alpha^2)}\theta_{x_n-1} \tag{24}$$

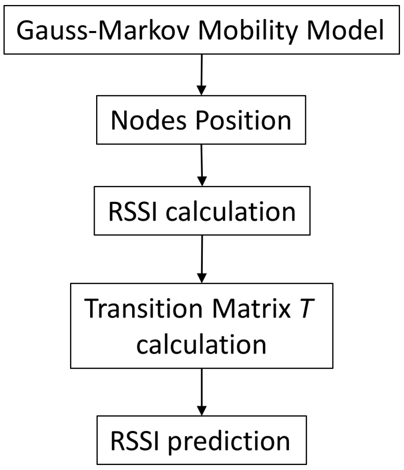


FIGURE 9. Mobility model.

where v_n and θ_n denote the new speed and direction of the mobile node at time interval n, respectively, $0 \leq \alpha \leq 1$ is the tuning parameter to vary the randomness degree, $\overline{v}$ and $\overline{\theta}$ are the expected values of the speed and direction as the Gauss–Markov random process, respectively, and v_{x_n-1} and θ_{x_n-1} are Gaussian distributed random variables with zero mean and unit variance, and independent of v_n and θ_n.

In this sense, positions provided by the mobility model were used to calculate distances, which were used to determine RSSI values, as follows [14]:

$$RSSI_d = RSSI_{d_0} - 10n * log_{10}\left(\frac{d}{d_0}\right) \tag{25}$$

In Equation 25, n, d_0 and $RSSI_{d_0}$ are given values, and configured for outdoor environments [14]. The first one corresponds to the path loss coefficient. The second one indicates a known distance of reference, and the third one establishes the RSSI level at distance reference d_0. Finally, the RSSI level for a specific distance d is calculated through the previously given data values.

As we presented in previous sections, these RSSI values were used as an indirect way (since we assume nodes are not equipped with GPS devices) to know how far are two pairs of nodes.

IV. IMPLEMENTATION AND RESULTS

Our mathematical optimization model was implemented using GAMS, and MATLAB was used for implementing the rest of the approaches. In order to properly understand the results, it is necessary to describe each label presented in the figures as follows:

- *Mathematical Model:* It corresponds to the multi-objective mathematical optimization proposed in section III-A. This mathematical model was implemented in GAMS. In addition, for this model, $w_1 = 0.5$ and $w_2 = 0.5$ to provide the same weight for the delay function and energy consumption function.
- *PAMC:* It corresponds to the Prediction Algorithm based on Markov Chains proposed in section III-B.

- *PADL for $n = 15$:* It corresponds to the Prediction Algorithm based on Deep Learning for $n = 15$ proposed in section III-C. Remember that $n = 15$ means that it will be considered the last fifteen values of distance and energy consumption of neighbour nodes to determine the best forwarding node.
- *PADL for $n = 10$:* It corresponds to the Prediction Algorithm based on Deep Learning for $n = 10$ proposed in section III-C. Remember that $n = 10$ means that it will be considered the last ten values of distance and energy consumption of neighbour nodes to determine the best forwarding node.
- *PADL for $n = 5$:* It corresponds to the Prediction Algorithm based on Deep Learning for $n = 5$ proposed in section III-C. Remember that $n = 5$ means that it will be considered the last five values of distance and energy consumption of neighbour nodes to determine the best forwarding node.
- *AODV for MWSN:* This algorithm corresponds to the traditional AODV algorithm enhanced to be applied in mobile wireless sensor networks [21]. Remember that this algorithm is based on distances to find routes and, thus, we want to know if our prediction algorithm can surpass it.
- *Random Algorithm:* This algorithm builds a path selecting a random neighbour node as a forwarding node.

These approaches (except the Mathematical Model) were tested 10000 times for each network size in order to obtain significant results. In other words, 10000 different scenarios for each network size were generated to evaluate each approach. The mathematical optimization model was tested 100 times for each network size instead of 10000 because it is an offline solution that requires too much time. In addition, the maximum number of network states (network movements) for each network size was 10000, that is, the number of times that each network node changed its position. This value was enough for finding a path from the source node to the sink for each approach.

In summary, we have proposed prediction distributed routing algorithms that takes into account the network mobility in order to build as fast as possible a path between a source node and a sink, comparing their performance against the optimal solution given by the mathematical optimization model and compared against traditional routing algorithms such as the AODV for MWSN and the Random Algorithm described previously.

Table 4 summarizes the most important parameters assumed in the simulations.

From Table 4, we assume to deploy the nodes in an area of $100 \times 100\ m^2$ considering a communication radius (r_c) of 20 meters. We also assume just one source node and one sink in order to build a path from this source node to the sink. The rest of the information is supported by references given in this table.

The metrics used to evaluate the performance of each approach are the following:

TABLE 4. Simulation parameters.

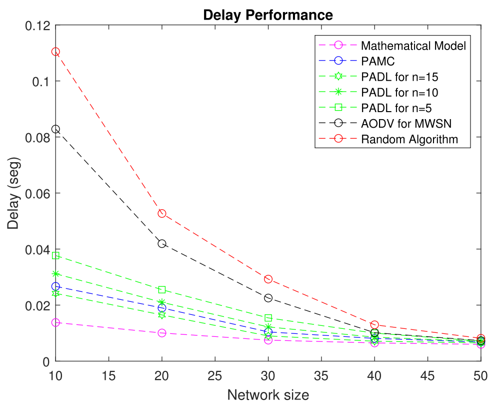
Parameters	Value
Work Area	100x100[m^2]
r_c	20[m]
E_{amp}	100[pJ/bit/m^2] [1]
E_{elec}	50[nJ/bit] [1]
Data packet size	128 bytes [22]
Control packet size	12 bytes [22]
Number of nodes (including the sink)	10-50
Number of sources nodes	1
Number of sinks	1
Number of simulations for each approach (For each network size)	10000

- *Delay:* It corresponds to the time needed to carry a data packet from the source node to the sink.
- *Energy Consumption:* It corresponds to the energy wasted for all the network since a data packet is transmitted from the source node until it is received by the sink.
- *Hops:* It corresponds to the number of hops taken by a data packet since it is transmitted from the source node until it is received by the sink.
- *Overhead:* It corresponds to the control packets required to build a path to carry a data packet from the source node to the sink.

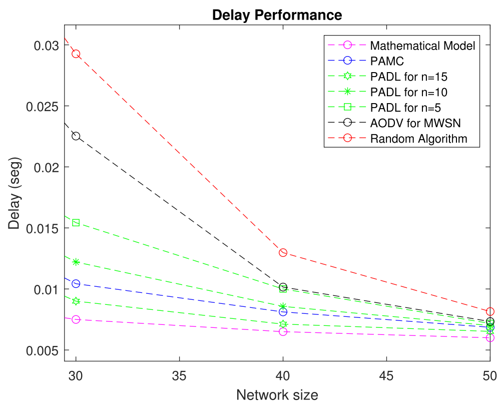
The most important results are described and analyzed in the following items:

- Figures 10a, 10c, 10e and 10g present the performance of each approach for delay, energy consumption, hops and overhead respectively. Figures 10b, 10d, 10f and 10h are just a zoom in version of Figures 10a, 10c, 10e and 10g respectively. This zoom in is performed to see in detail the performance from 30 to 50 nodes.
- In all figures, we could verify that our mathematical model always obtained the best results, which was expected. In other words, our mathematical model proposal always obtains the best solution for each network size for all metrics. Remember that the mathematical model needs too much time to provide a solution, whereby it is not an affordable solution for real mobile wireless sensor networks applications because they require solutions that must be obtained as fast as possible. In this sense, in this work our mathematical optimization model is used as an offline method that has global information about the network, which allows us to obtain the best possible solution, that is, the optimal solution value for a specific network scenario. For this reason, it is obvious that the mathematical model always obtains the best results for all metrics evaluated. As a consequence, the optimal solutions offered by the mathematical model can be used as reference values to evaluate how close is the performance of our prediction algorithms (PAMC and PADls), AODV for MWSN, and the Random Algorithm to these optimal solutions.

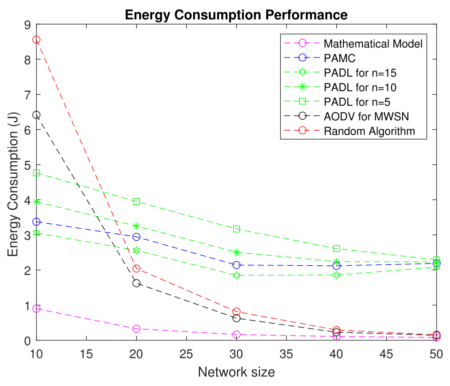
- Figures 10a and 10b show the delay performance of all approaches for each network size (10 to 50 nodes). Here, our prediction algorithms (PAMC and PADLs), after the mathematical model, showed a better performance than AODV for MWSN and Random Algorithm for each network size. These results confirm that using mobility prediction is very useful to establish fastly a path from a source node to the sink. In detail, in terms of the Deep Learning approach, the model used for $n = 15$ (PADL for $n = 15$) showed better results than the Markov Chains approach (PAMC). In other words, after the mathematical model, the PADL for $n = 15$ obtained the best results in terms of delay performance. This indicates that a high value of n (15) for PADL allowed us to predict more precisely the movement pattern of neighbour nodes and, thus, selecting a better forwarding node. In other words, a higher value of n allows the learning model (PADL) to have more distance information in order to improve the distance prediction of its neighbours. Remember that a better forwarding node represents the node that will be closer in the future to the node i, that is, the node that has a data packet and is deciding which neighbour node must be selected as a forwarding node to build a path between a source node and the sink. Selecting a forwarding node with this method reduces the delay caused for communication disruptions.
  On the other hand, as network size decreases, the delay performance of our prediction algorithms (PAMC and PADLs) is each time better than AODV for MWSN and the Random Algorithm. This means that if our network has few nodes, our prediction algorithms (PAMC and PADLs) are capable of obtaining a large advantage against the other algorithms (AODV for MWSN and the Random Algorithm) for finding the destination node. Few nodes mean that there is less probability to find a neighbour node, and thus, is more difficult to establish a path to the sink. However, our prediction algorithms are capable of finding reliable forwarding neighbour nodes, allowing us to establish fastly a communication path to the sink. In other words, the fact of finding reliable forwarding neighbour nodes indicates we are selecting forwarding nodes with less probability of suffering an interruption, that is, neighbour nodes that, in the next future, will be closer to the node that currently has a packet to be sent.
  In addition, as network size increases, the delay performance of AODV for MWSN and the Random Algorithm is each time closer to our prediction algorithms (PAMC and PADLs). This means that if a network has many nodes, this favors the fact of establishing a path for AODV for MWSN and the Random Algorithm. Many nodes mean that there is more probability to find a neighbour node, and thus, is easier to establish a path to the sink since there is less chance to suffer an interruption. In summary, the prediction capability is more effective as network size decreases.
  Finally, in terms of delay performance, it is recommended to use our prediction algorithms as network size decreases, that is, for small networks (10 to 40 nodes). In contrast, as network size increases, the prediction capability begins to be irrelevant against traditional solutions such as AODV for MWSN and the Random Algorithm.
- Figures 10e and 10f show the hops performance of all approaches for each network size (10 to 50 nodes). There is a clear proportionality between the delay performance and the hops performance. This is because the more hops are necessary to traverse the network to finally achieve the sink, the more delay is required. In this sense, the behavior of each approach for the hops performance evaluation is equivalent to the delay performance. For this reason, the same analysis done for the delay performance is applied to the hops performance.
- Figures 10g and 10h show the overhead performance of all approaches for each network size (10 to 50 nodes). According to these figures, our prediction algorithms (PAMC and PADLs) require many control packets to build as fast as possible a path between a source node and the sink. The more nodes are in the network, the more control packets are needed for our prediction algorithms because each network node continually collects distance information from its neighbour nodes according to the information provided in sections III-B and III-C.
- Figures 10c and 10d show the energy consumption performance of all approaches for each network size (10 to 50 nodes). According to these figures, for 10 and approximately 15 nodes in the network, our prediction algorithms (PAMC and PADLs) have less energy consumption than AODV for MWSN and the Random Algorithm. In detail, PADL for $n = 15$ obtained the best results in terms of energy consumption performance for 10 and approximately 15 nodes in the network. PADL for $n = 15$ needed less energy consumption than the other algorithms (not the mathematical model) because it required a less number of hops and control packets to build a path between a source node and the sink. That is, the fewer hops and control packets are needed to build a path, the fewer amount of transmission and reception processes that waste energy are needed. On the other hand, as network size increases, the energy consumption of our prediction algorithms (PAMC and PADLs) increases because the more nodes the network has, the more control packets are needed to build a path. Remember that, in our prediction algorithms, all nodes continually collect distance information from their neighbour nodes to build a path to the sink. As a result, the energy consumption of AODV for MWSN and the Random Algorithm is less than our prediction algorithms because



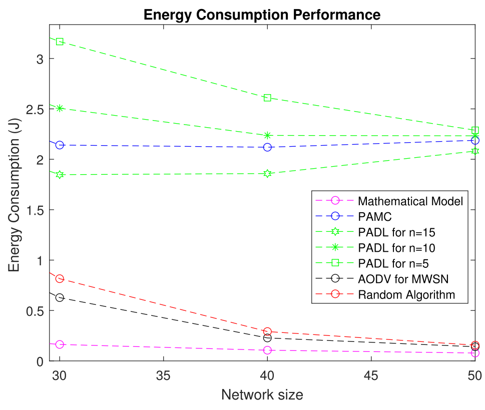
(a) Delay Performance.



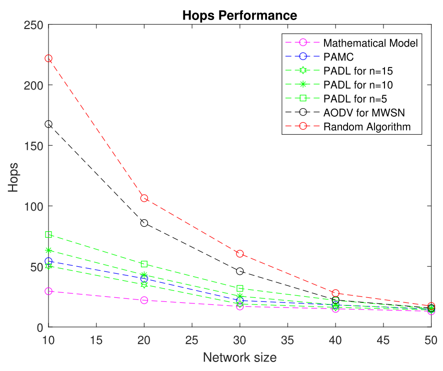
(b) Zoom for Delay Performance.



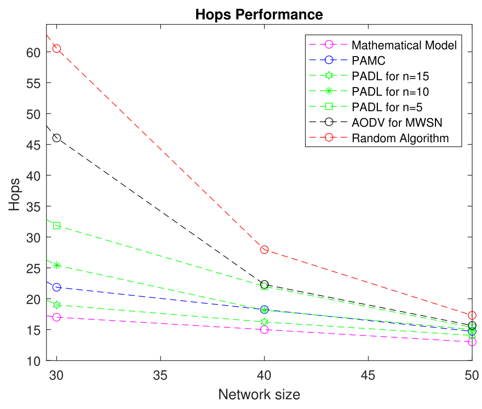
(c) Energy Consumption Performance.



(d) Zoom for Energy Consumption Performance.



(e) Hops Performance.

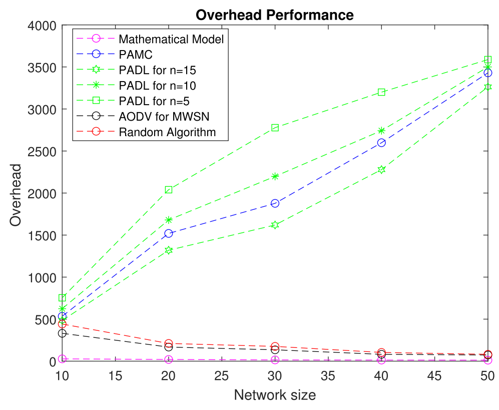


(f) Zoom for Hops Performance.

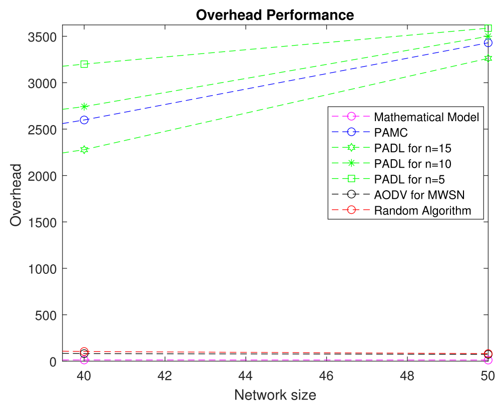
FIGURE 10. **Results.**

they do not need as many control packets as our prediction algorithms.

In summary, the performance of our prediction algorithms (PAMC and PADLs) in small networks (10 to 40 nodes) is very beneficial since it allows us to find fastly a path between a source node and a sink. However, as network size increases, the prediction capability begins to be irrelevant against traditional algorithms such as AODV for MWSN and the Random Algorithm. However, analyzing at the same time the delay and the energy consumption performances, our prediction algorithms are recommended to be used in small networks, that is, from 10 to 15 nodes approximately. Anyway, our prediction algorithms offer the best results in terms of delay as the network size increases, specially PADL for $n = 15$, but at the expense of increasing the energy consumption of the network. Thus, our prediction algorithms could be used depending on the requirements of the application. For example, we can simply use AODV for MWSN or the Random Algorithm for applications that do not have delay



(g) Overhead Performance.



(h) Zoom for Overhead Performance.

FIGURE 10. ***(Continued.)*** **Results.**

requirements but needs to minimize energy consumption. On the other hand, if the application has delay requirements but does not need to minimize the energy consumption, we can use our prediction algorithms, specially PADL for $n = 15$. Finally, if the application has requirements in terms of delay and energy consumption, we can use our prediction algorithms in small networks, but if the network size is not small, we can use our prediction algorithms at the expense of increasing the energy consumption of the network.

V. CONCLUSION

We proposed a multi-objective optimization model and prediction distributed routing algorithms based on Markov Chains and a Deep Learning approach for finding the minimum cost path between a source node and a gateway node (destination node) considering all nodes are mobile. The results obtained by the mathematical optimization model served as a reference to evaluate our prediction algorithms and other traditional algorithms in order to analyze their performance in terms of delay and energy consumption in MWSN. In other words, optimal values given by the mathematical model were be used to determine how good were the results obtained by the algorithms. Additionally, we implemented typical distributed routing algorithms to know the performance of the prediction distributed routing algorithm. As expected, our mobility prediction algorithms obtained the best solutions in terms of delay and energy consumption compared against not-using prediction techniques (AODV for MWSN and the Random Algorithm), being more effective as network size decreases.

In detail, our mobility prediction algorithms allowed us to establish the most reliable path for finding the sink and, at the same time, it allowed us to obtain the best path to the sink compared against traditional algorithms in terms of delay and energy consumption. Thus, the reliability offered by the mobility prediction algorithms allowed us to select the most stable forwarding nodes in terms of their network connectivity. In this sense, it was less likely that a data message would be in isolated network zones, and then, there was a higher probability to reach the sink by the data message. For this reason, when the number of network nodes was scarce, the mobility prediction algorithms performance was too high in terms of delay in comparison with the rest of the algorithms. In other words, we proposed to apply our prediction algorithm in networks of 10 to 40 nodes because we considered more interesting the fact of applying the prediction methods in scarce networks where the number of neighbours is very limited and, for this reason, the probability of finding a sink is much less than in large networks. This means that if our network has few nodes and, as a consequence, it is more difficult to find a path to a sink, our prediction algorithm was capable to obtain a large advantage in terms of delay against traditional algorithms for finding the sink.

In terms of energy consumption, the energy performance of our proposal besides the delay performance makes the mobility prediction algorithms totally suitable for scarce networks, that is, for mobile wireless sensor networks applications where the number of nodes is not too high and it is required data messages arrive at the sink as soon as possible.

In summary, the performance of our prediction algorithms in small networks (10 to 40 nodes) is very beneficial since they allow us to find fastly a path between a source node and a sink at minimum energy consumption. However, as network size increases, the prediction capability begins to be irrelevant against traditional algorithms such as AODV for MWSN and the Random Algorithm.

As future works, we are planning to evaluate our dataset in other network simulators such as OMNET++ and EDGF [30]. On the other hand, our proposed network simulator was developed taking into account the network layer, but, in addition, we are considering to extend our proposal for incorporating the MAC layer.

REFERENCES

[1] I. F. Akyildiz and M. C. Vuran, *Wireless Sensor Networks*. Hoboken, NJ, USA: Wiley, 2010.
[2] J. Zheng and A. Jamalipour. *Wireless Sensor Networks: A Networking Perspective*. Hoboken, NJ, USA: Wiley, 2009.
[3] A. A. Ahmed, "An enhanced real-time routing protocol with load distribution for mobile wireless sensor networks," *Comput. Netw. J.*, vol. 57, no. 6, pp. 1459–1473, 2013, doi: 10.1016/j.comnet.2013.02.003.

[4] A. A. Ahmed, *Real-TimeWireless Sensor Networks*. Charlottesville, VA, USA: Univ. Virginia, 2007.
[5] B. Buchli, F. Sutton, and J. Beutel, "GPS-equipped wireless sensor network node for high-accuracy positioning applications," in *Wireless Sensor Networks* (Lecture Notes in Computer Science), vol. 7158. Zürich, Switzerland: ETH Zürich, 2012, pp. 179–195.
[6] S. Li, X. Ma, X. Wang, and M. Tan, "Energy-efficient multipath routing in wireless sensor network considering wireless interference," *J. Control Theory Appl.*, vol. 9, no. 1, pp. 127–132, Feb. 2011.
[7] G. M. de Araújo, A. R. Pinto, J. Kaiser, and L. B. Becker, "Genetic machine learning approach for link quality prediction in mobile wireless sensor networks," in *Cooperative Robots and Sensor Networks*. Berlin, Germany: Springer-Verlag, 2014.
[8] G. M. de Araujo, J. Kaiser, and L. B. Becker, "An optimized Markov model to predict link quality in mobile wireless sensor networks," in *Proc. IEEE Symp. Comput. Commun.*, Jul. 2012, pp. 307–312, doi: 10.1109/ISCC.2012.6249313.
[9] J. A. Torkestani, "Mobility prediction in mobile wireless networks," *J. Netw. Comput. Appl.*, vol. 35, no. 5, pp. 1633–1645, Sep. 2012, doi: 10.1016/j.jnca.2012.03.008.
[10] *A Community Resource for Archiving Wireless Data at Dartmouth*. Accessed: Mar. 5, 2021. [Online]. Available: http://crawdad.org/
[11] M. A. Habib, S. Saha, M. A. Razzaque, M. Mamun-or-Rashid, G. Fortino, and M. M. Hassan, "Starfish routing for sensor networks with mobile sink," *J. Netw. Comput. Appl.*, vol. 123, pp. 11–22, Dec. 2018, doi: 10.1016/j.jnca.2018.08.016.
[12] A. Kaswan, V. Singh, and P. K. Jana, "A multi-objective and PSO based energy efficient path design for mobile sink in wireless sensor networks," *Pervasive Mobile Comput.*, vol. 46, pp. 122–136, Jun. 2018, doi: 10.1016/j.pmcj.2018.02.003.
[13] N. N. Srinidhi, S. M. D. Kumar, and K. R. Venugopal, "Network optimizations in the Internet of Things: A review," *Eng. Sci. Technol., Int. J.*, vol. 22, no. 1, pp. 1–21, Feb. 2019, doi: 10.1016/j.jestch.2018.09.003.
[14] C.-C. Pu, C.-H. Pu, and H.-J. Lee, "Indoor location tracking using received signal strength indicator," in *Emerging Communications for Wireless Sensor Networks*. London, U.K.: InTech, 2010, doi: 10.5772/10518.
[15] G. A. Montoya and Y. Donoso, "A prediction algorithm based on Markov chains for finding the minimum cost path in a mobile WSNs," *Int. J. Comput. Commun. Control*, vol. 14, no. 1, pp. 39–55, Feb. 2019, doi: 10.15837/ijccc.2019.1.3487.
[16] A. Yousefiankalareh, A. Najari, and M. Hosseynzadeh, "Tree-based routing protocol in wireless sensor networks using optimization algorithm batch particles with a mobile sink," in *Proc. IEEE 17th Int. Conf. Smart Communities, Improving Qual. Life ICT, IoT AI*, Dec. 2020, pp. 1–5, doi: 10.1109/HONET50430.2020.9322844.
[17] Y. Hu, Y. Zheng, H. Liu, Z. Wang, Y. Mao, and H. Han, "Mobile sink path planning research for underwater heterogeneous sensor network," in *Proc. Chin. Control Decis. Conf.*, Jun. 2018, pp. 4443–4448, doi: 10.1109/CCDC.2018.8407899.
[18] X. Li and J. Guan, "SORA: A stochastic optimal routing algorithm for wireless sensor networks," in *Proc. IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun.*, Oct. 2017, pp. 1–6, doi: 10.1109/PIMRC.2017.8292491.
[19] J. Kumar, S. Tripathi, and R. K. Tiwari, "Routing protocol for wireless sensor networks using swarm intelligence-ACO with ECPSOA," in *Proc. Int. Conf. Inf. Technol. (ICIT)*, Dec. 2016, pp. 23–27, doi: 10.1109/ICIT.2016.018.
[20] Vinamrata, R. Paulus, A. K. Jaiswal, and M. Kumar, "MEACSRA: Mobility and energy-aware cross-layer searching routing algorithm for wireless sensor network," in *Proc. 5th Int. Conf. Rel., Infocom Technol. Optim. (ICRITO)*, Sep. 2016, pp. 547–552, doi: 10.1109/ICRITO.2016.7785016.
[21] M. Rajesh, K. Vanishree, and T. S. B. Sudarshan, "Stable route AODV routing protocol for mobile wireless sensor networks," in *Proc. Int. Conf. Comput. Netw. Commun.*, 2015, pp. 917–923, doi: 10.1109/CoCoNet.2015.7411300.
[22] S. Kurt, H. U. Yildiz, M. Yigit, B. Tavli, and V. C. Gungor, "Packet size optimization in wireless sensor networks for smart grid applications," *IEEE Trans. Ind. Electron.*, vol. 64, no. 3, pp. 2392–2401, Mar. 2017, doi: 10.1109/TIE.2016.2619319.
[23] J. Ding, H. Liu, L. T. Yang, T. Yao, and W. Zuo, "Multiuser multivariate multiorder Markov-based multimodal user mobility pattern prediction," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4519–4531, May 2020, doi: 10.1109/JIOT.2019.2951134.
[24] H. Wang, Y. Li, D. Jin, and Z. Han, "Attentional Markov model for human mobility prediction," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2213–2225, Jul. 2021, doi: 10.1109/JSAC.2021.3078499.
[25] M. Yan, S. Li, C. A. Chan, Y. Shen, and Y. Yu, "Mobility prediction using a weighted Markov model based on mobile user classification," *Sensors J.*, vol. 21, no. 5, p. 1740, 2021, doi: 10.3390/s21051740.
[26] X. Liu, J. Yu, H. Qi, J. Yang, W. Rong, X. Zhang, and Y. Gao, "Learning to predict the mobility of users in mobile mmWave networks," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 124–131, Feb. 2020, doi: 10.1109/MWC.001.1900241.
[27] Z. Fan, X. Song, T. Xia, R. Jiang, R. Shibasaki, and R. Sakuramachi, "Online deep ensemble learning for predicting citywide human mobility," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 2, no. 3, pp. 1–21, Sep. 2018, doi: 10.1145/3264915.
[28] Y. Yayeh, H.-P. Lin, G. Berie, A. B. Adege, L. Yen, and S.-S. Jeng, "Mobility prediction in mobile ad-hoc network using deep learning," in *Proc. IEEE Int. Conf. Appl. Syst. Invention (ICASI)*, Apr. 2018, pp. 1203–1206, doi: 10.1109/ICASI.2018.8394504.
[29] A. S. A. Sukor, L. M. Kamarudin, A. Zakaria, N. A. Rahim, S. Sudin, and H. Nishizaki, "RSSI-based for device-free localization using deep learning technique," *Smart Cities*, vol. 3, no. 2, pp. 444–455, Jun. 2020, doi: 10.3390/smartcities3020024.
[30] D. K. Sah, K. Cengiz, P. K. Donta, V. N. Inikollu, and T. Amgoth, "EDGF: Empirical dataset generation framework for wireless sensor networks," *Comput. Commun. J.*, vol. 180, pp. 48–56, Dec. 2021, doi: 10.1016/j.comcom.2021.08.017.

GERMÁN A. MONTOYA received the degree in electronics engineering from Universidad Pontificia Bolivariana, Medellín, Colombia, in 2006, and the Ph.D. degree in engineering from the Universidad de los Andes, Bogotá, Colombia, in 2017. Currently, he is a Postdoctoral Assistant with the System and Computing Engineering Department, Universidad de los Andes. His main research interests include mathematical optimization, simulation, and wireless sensor networks.

CARLOS LOZANO-GARZON (Senior Member, IEEE) received the System Engineering degree from the Universidad Nacional de Colombia, Bogotá, Colombia, in 2002, and the joint Ph.D. degree from both the Universidad de los Andes, Colombia, and the Universitat de Girona, Spain, in 2017. He is currently an Assistant Professor with the System and Computing Engineering Department, Universidad de los Andes. He worked as a Professor at the Universidad Nacional de Colombia, among others, and as a Visiting Professor at the Universidad Nacional de Asunción, Paraguay. His research interests include network optimization from a multi-criteria perspective, smart mobility: vehicle networks and intelligent transport systems, the Internet of Things, and network security.

YEZID DONOSO (Senior Member, IEEE) received the System and Computing Engineering degree from the Universidad del Norte, Barranquilla, Colombia, in 1996, the M.S. degree from the Universidad de los Andes, Bogotá, Colombia, in 1998, and the D.E.A. and Ph.D. *(cum laude)* degrees in information technology from the University of Girona, Girona, Spain, in 2002 and 2005, respectively.

He is currently a Full Professor with the System and Computing Engineering Department, Universidad de los Andes. He is the coauthor of several books, including *Multi-Objective Optimization in Computer Networks Using Metaheuristics* and *Network Design for IP Convergence*. He has more than 240 papers published between journals and conferences. He has been a Senior Member of the IEEE Computer Society Distinguished Visitors Program (DVP), since 2005, and was the IEEE Colombia Section Chair, from 2013 to 2014. He has several national and international awards and medals.