JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

An intelligent PRoPHET - Optimized Random forest-based multi-copy routing algorithm for opportunistic IoT networks

Vaishak D A, Vaibhav D A
B.E. Students
Dept of Computer Science and Engineering
UVCE Bangalore, India
BMSCE Bangalore India

***Abstract*—Opportunistic networks remain one of the essential categories of ad hoc networks in the Internet of Things (IoT), which examines individual activities like regular routines, ventures, and various more to present effective communication. In opportunistic networks, movable nodes stabilize the connection between nodes despite the non-availability of a dedicated route. Moreover, nodes do not gain any information in advance about the aspects of the network, such as the network topology and the position of the other nodes. Consequently, devising a routing algorithm becomes a demanding task as conventional routing protocols used in the Internet are infeasible for the characteristics of in- herent type of network. The suggested work propounds a multi-copy routing algorithm based on machine learning named i*PRoPHET* or intelligent PRoPHET (Probability routing protocol utilizing records of encounters and transitivity). i*PRoPHET* uses effective chang- ing contextual data of nodes and the transmission probability of PRoPHET to carry out information transfer. The i*PRoPHET* uses a machine-learning-based classifier known as random forest to divide the node as a reliable forwarder or a non-reliable forwarder based on the sup- plied contextual data during each routing decision. The classifier trained with a considerable measure of data derived utilizing simulation drives accurate classification as reliable or unreliable nodes for carrying out the routing task. The results obtained from the simulation prove that the proposed algorithm outperforms concerning delivery probability, hop count, overhead ratio, and latency but over costs concerning average buffer time in par with the same multi-copy routing algorithms. The uniqueness of this paper lies in data extraction, categorization, and training the model to obtain reliable and unreliable nodes to facilitate efficient multi-copy routing in IoT communication.**

Keywords*— *Internet of things; Machine learning; Mobility; Multi-copy; Random forest.

I. INTRODUCTION

As technology advances, more devices are connected to the Internet, resulting in new networking and the communication criterion known as the Internet of Things (IoT). Therefore, many devices are being monitored and are no longer able to be deployed statically. These IoT devices will interact statically or dynamically while being transported by humans or different carriers. For example, in a vehicular-based IoT network, statically placed devices may need to detect mobile agents and use them to compile and send their data to the desired destination. Opportunistic mobile networks have lately emerged as a novel transmission method in wireless systems. Unlike MANETs(mobile ad hoc networks), which need end-to-end linked pathways for information transfer,

opportunistic mobile networks rely on the collaboration of opportunistic connections among mobile nodes without the availability of end-to-end communication links. When people associate with these mobile devices, they form connections, and these network types are referred to as human social networks. As a result, opportunistic networks use individual actions and social relationships to create more effective and dependable message delivery systems. Data gathering, propagation, and dissemination generally need opportunistic connections employing mobility, short-range transmission, and technologies referred to as OppIoT [1]. During such processes, routing protocol configuration pretends difficulty due to its intrinsic complexity in acquiring connectivity among devices and identifying an appropriate forwarder to forward the data packet to their target destination. Opportunistic networks [2], a derivative class of OppIoT, are an advanced version of MANETs.In OppNets, nodes are permitted to establish communication between various other nodes even though they do not have a routing connection.

Nodes are connected temporarily, and the routes keep changing dynamically due to the node's mobility and the node's frequent deactivation and activation. When packets are en route between the source and target nodes, any feasible node in the middle can be used as the next-hop whenever it is plausible to deliver the packets to the intended destination with the shortest possible routing distance. In Opportunistic routing, the message will be propagated in two ways as single-copy and multi-copy. Single-copy routing has a lower delivery probability since only one copy of the message exists in the network, and the intended recipient may fail to get the message if it is dropped; examples of single-copy routing are First Contact, Direct Transmission/Delivery, and so on. In multi-copy, many copies of the same message exist in the network to increase the probability of delivery. Multi-copy is moreover divided into two groups as unlimited copy and n-copy routing protocols [3].In n-copy routing, only a specific and configurable number of message copies are kept in the network. In contrast, in unlimited copy routing, messages are transmitted without any limitations on message creation, resulting in higher network packet overhead.

Spray and Wait and PRoPHET are examples for n-copy routing, and Probabilistic, epidemic routing are examples for unlimited copy routing protocols [4]. The suggested method employs n-copy routing, with three data copies maintained in the network. Several protocols about OppNets have been postulated, like an epidemic, HBPR, and HibOp, which use parameters such as message transmission pattern, node context erudition (i.e., present position, delivery possibility, target node proximity, previous and current history of the node), last synergetic time, and various other variables to predict which node is chosen as the next hop to deliver the message to the designed destination. This paper proposes a machine learning-based technique that may be used for OppNets. The proposed approach differs significantly from the existing algorithm. It uses a random forest to choose the adjacent node to which the data packet should be sent based on the current context and position information. Random forest, often known as random decisions forest, is one of the most widely used ensembles learning approaches in classification and regression. During the training stage, multiple decision trees are constructed at random with various subsets of data. The mode determines which class is allocated. The suggested iPRoPHET's random forest algorithm employs the bootstrap aggregation approach, which improves the system's stability and accuracy while also lowering variance and overfitting. OppIoT nodes are vulnerable to attacks from malicious nodes masquerading as IoT nodes [5].Authentication architecture [6], which utilizes a smart card to ascertain node authenticity, dummy location privacy-preserving algorithm [7], and safeguarding the location privacy of mobile device users in location-based services [8], can all be used in the proposed work to assist nodes in the OppIoT network in hiding their location and providing dummy data to malicious nodes to prevent them from joining the network.

The rationale of the proposed work is (1) The dynamic nature of nodes in an opportunistic network over time makes it difficult to route information and identify intermediary nodes utilized to forward messages to their destinations. (2) To improve the efficiency of the multi-copy routing algorithm's network parameters. (3) Machine learning classifiers are used to categorize nodes and find possible forwarder nodes. (4) Improving the classification level by combining existing historical data with current PRoPHET algorithm context information.

The suggested work's contribution is

• Improving the preexisting iPRoPHET algorithm's IoT network characteristics such as delivery probability, hop count, overhead ratio, and latency.

• During each routing choice, identifying intermediate forwarder nodes as reliable or unreliable depends on contextual information.

• The existing PRoPHET algorithm does not use historical data and instead relies solely on current contextual information in its static model. As a result, the iPRoPHET algorithm classifies forwarder nodes based on both previous and present context information.

• Random forest is utilized in the proposed approach, which employs the ensemble learning technique, and the model is made robust by generating several trees and aggregating them.

• When compared to their counterparts, the use of random forest in the suggested iPRoPHET provides better classification because it uses boosting and bagging

techniques and helps in avoiding overfitting. Random forest classifier trained with large amounts of data obtained using simulation leads to precise node classification.

The paper is organized as follows: Section 2 summarizes several existing research on multi-copy and opportunistic networks. Section 3 presents the problem statement. In Section 4, the proposed algorithm is provided. In Section 5, the experimental assessment and findings are discussed. Section 6 contains the conclusion as well as recommendations for further study.

II. RELATED WORK

Many of the protocols used in OppNets are concerned with node context and message delivery. The following are the summaries of the representative ones. Epidemic routing [9] uses a flood-based routing method whose primary goal is to deliver a packet with a high probability to the intended destination while requiring the least amount of information about the topology and network. The approach is built on an epidemic algorithm that handles two buffers: one for messages created by the node and the other for messages generated by various nodes. A summary vector present in each node and includes a brief description of the messages presently cached in the buffer. Nodes modify their summary vectors when they come into touch with one another, and each node includes an additional buffer to prevent verbose connections with other nodes. The approach is built on an epidemic algorithm that handles two buffers: one for messages created by the node itself and the other for messages generated by various nodes. After a certain amount of time, nodes trade their summary vectors with each other, and after exchanging their vectors, nodes compare their vectors to see which message is missing. The nodes then request copies of the messages that they don't have. On the other hand, the postulated epidemic protocol requires more bandwidth and buffer space, causing network congestion, and it does not look at node characteristics to forecast message delivery. Dhurandher et al. [10] introduced the HBPR (History Depending Prediction for Routing), which chooses the intermediate node as the message carrier based on specific characteristics. The period for which a pair of nodes encounter, the node's history, the moment at which nodes meet, and the path of the node's mobility between the root and the target node are the parameters. The schema is divided into three stages: identifying the home location, when the motion of the node is used to determine the next mode location; message production, wherein the intermediate node is decided, and the home location is renewed; and finally, the resulting phase for selecting hop, during which metric is used to judge on the assortment. Compared to Epidemic, Protocol outperformed Epidemic in terms of message delivery quantity and overhead ratio, but only after considering the individual mobility paradigm. In an actual application setting, the PRoPHET routing protocol [11] employs non-arbitrary motion and association patterns to repeat the message to multiple nodes to improve routing performance. It is calculated on the assumption that if a node has interacted with another node or visited a location on a frequent basis, there is a greater chance that the region or node will be revisited. A delivery probability is defined for each node that associates with other nodes to do this. Assume that a node T intends to send a packet to node U; if this node T is associated with another node R, which has a better delivery probability to node U than T, a copy of the payload is copied to node R. The buffer size aggregate, popularity, position, and predictability of delivery are used to determine deliverability. If a source node interacts with several nodes simultaneously, the message is replicated to the node with the most excellent deliverability value. The routing algorithm distance is utilised as an additional element in distance-based PRoPHET. Each packet-carrying node calculates the distance to all neighbouring nodes and selects the closed node as packet forwarders. As a result, the probability of delivery increases while the likelihood of delay decreases. Dhurandher et al. [12] introduced the Prowait routing algorithm, which combines the PRoPHET and Spray and waits for routing protocols. When two nodes engage in communication, packets are transmitted from the node with a lower likelihood of delivery to the node with a greater probability of delivery to the target destination. During packet transmission, they primarily consider a few factors. 1) selection of a neighbouring node 2) selecting the next-hop node. The PRoPHET protocol is used to assess delivery predictability during node selection, and the data is sent via the spray and wait for the routing protocol. The PRoPHET routing method is used to choose the node for the next hop. The duplicate data packets are kept by the Prowait routing algorithm, which ensures that the packets arrive at their intended destination with the least amount of delay. When compared to the epidemic procedure, it lowers resource waste by depreciating the quantity of flooding. However, it does not anticipate message packet forwarding to reduce network delivery delays. Boldrini et al. [13] introduced the HiBO protocol (history-based routing protocol for OppNets), using current context statistics about nodes to choose the best node to transmit the message to the intended destination. The simulation environment is used to acquire the context statistics. When a node wishes to send a message, it uses the tables to determine if a union exists between the context statistics of its surrounding nodes and the message data. Data packets are forwarded to the next node, where the union coexists in this scenario. This routing algorithm will take longer to determine which node is the adjacent node and demand more work to maintain the tables. The EDR

(encounter and distance-based routing algorithm) [14] protocol transmits messages from the source to the destination using context statistics. The routing protocol determines the contact value for each pair of nodes (i.e. the frequency at which they encounter) dynamically. Furthermore, the Euclidean distance is calculated dynamically for each pair of nodes. Based on the values discussed before, a parameter called the best-forwarder is assessed, and the packet is only sent to those intermediate nodes whose forwarder value is greater than the threshold. Because it is a context-dependent protocol, it performs poorly in terms of packet delivery probability and overhead ratio. Jeon and Lee [15] presented a Markov chain prediction model in which a particular node's mobility behaviour is described based on the node's previous mobility information state to forecast the node's predicted movement. The packet delivery ratio of mobile nodes to destination nodes is determined using this data. In comparison to traditional routing methods for OppNets, the suggested approach improves delivery probability and lowers latency. However, this protocol falls short compared to alternative mobility models, such as a random waypoint. MLProph [16], an infrastructure-free OppNet routing system, employs a machine learning technique to determine prior network knowledge and node characteristics such as buffer size, hop-count, delivery, and momentum. The chance of a node acting as a forwarder is calculated using these parameters. However, it ignores the node's message delivery probability, which is an important metric to consider while routing.

However, in the suggested model, a random forest method is used to select an adjacent node to which the data packet should be routed depending on the current context and location information. During the training stage, multiple decision trees are built at random using various subsets of data, which helps to improve the algorithm's stability and accuracy while decreasing variance and overfitting.

III. PROBLEM STATEMENT AND CHALLENGES OF THE PROPOSED WORK

3.1. Challenges

The opportunistic network in IoT confronts several problems. This is due to the changing nature of nodes over time dependent on the application, which makes routing information and identifying intermediary nodes utilized for message forwarding to destinations difficult.

• In the OppIoT, data gathering, propagation, and dissemination are generally accomplished through opportunistic connections that use mobility, short-range transmission, and technology. Due to the inherent difficulty of establishing communication across devices and detecting a relevant for- warder to forward data packets to their target destination during such activities, offers a problem to routing protocol configuration.

• Data extraction: An extensive data collection must be gathered to train the machine learning-based random forest model. It wasn't easy to get data that was near to the real world. The closest solution is the ONE simulator [17], and to retrieve the data from the ONE simulator, the source code must be modified to train the model.

• Model Creation: After collecting data from the ONE simulator, a reliable random forest classifier had to be built. To accomplish so, features and algorithm-related parameters must be carefully selected.

• Model Integration: To categories the nodes, the model must be correctly integrated with the ONE simulator after it has been built.

3.2. Problem statement

The following is the problem statement for the proposed work.

1. During each routing decision, identify the node as a trustworthy or unreliable forwarder based on the provided contextual information to maximize delivery probability, overhead ratio, hop count, and latency of the IoT connection.

2. To optimize the classification of nodes as dependable or unreliable nodes for carrying out routing information by training the random forest classifier with a huge quantity of data taken from the simulator.

IV. PROPOSED ALGORITHM

This section provides a detailed explanation of the proposed iPRoPHET algorithm.

4.1. Terminology

• S: Source node

• D: Destination node

• X: Chosen forwarder node

• G: Set of all adjacent nodes

• Ni: Neighbouring node i in the set S

• Mi: Message

• n: Prevalent node, responsible for routing decisions

• M: Set of all messages created during the data creation phase

• Ms: Set of messages transmitted

• Md: Set of dropped or aborted messages

• E: Set of all events collected during PRoPHET's data extraction phase

• P (Ni): PRoPHET-calculated delivery probability for the neighbour node using the current node n, Ni

• feature_name(x): In any set, the value of the attribute feature name in the record x.

4.2. System model

Consider N nodes creating an opportunistic network with communication capabilities and sufficient energy. Every node has enough buffer to hold state information such as the node's present position, delivery probability, direction, distances between source and destination, history, and power status. All of these characteristics may be utilized to identify the message transmission path. The PRoPHET is an n-copy routing method from the multi-copy routing category that calculates the delivery probability between nodes based on prior contact history. Even though this approach gives an adequate performance, machine learning models might improve selecting the next node. The PRoPHET does not fully exploit all of the context information available in the nodes. It is feasible to find the best predictors/features and utilize them to drive routing decisions using a data-driven strategy. Random forest algorithm is utilized in the suggested Random Forest-Based Optimized Multi-Copy Routing for Opportunistic IoT Networks to pick the adjacent node to route the datagram based on the current context and location information .

Random forests are one of the most often used ensemble learning algorithms for classification and regression. During the training stage, many decision trees are built at random using various subsets of data. The mode determines which class is allocated. The random forest method employs the bootstrap aggregation approach, which aids in increasing the system's stability and accuracy while minimizing variance and overfitting. Consider a training set X = x1,..., xn and their associated outputs Y = y1,..., yn, resulting in a dataset Dn = (X1, Y1),..., yn (Xn, Yn). Random forest comprises different base regression trees rn(x, m, Dn), m 1, where 1, 2, etc., are randomizing variable outputs

$$\bar{r}n(X,D_n)=E_\theta[rn(X,\theta,D_n)] \quad (1)$$

Where Eθ denotes the expectation in relation to θ, based on the randomizing variable theta, the dataset is split into multiple subspaces. It's also utilized to figure out how the individual trees are built and how the subsequent cuts are made. Feature bagging is when the algorithm chooses a random selection of features throughout the learning phase. If a characteristic is a good predictor, it will appear multiple times in various trees. Cross-validation techniques are used to determine the appropriate number of trees. The major benefit of Random forest over traditional decision trees is that single tree forecasts are subject to noise. However, the forest's aggregation will offset this disadvantage. Weighted neighbourhood schemes are another name for the random forest method. Based on contextual information, the random Forests method is used to categorise surrounding nodes as dependable or unreliable forwarders. The key steps of our suggested algorithm are as follows: (1) Gathering data from a single simulator (2) constructing and training the model.

4.2.1 . Stage 1: collecting routing data from one simulator

One of the finest simulation environments for modelling opportunistic mobile networks is One Simulator [20]. Modifications to one simulator source code are made at this stage to retrieve the information needed to train the algorithm. They have created a total of three datasets.

1. Event Dataset: This dataset records every event that occurred within the simulation environment. In a discrete-event simulation system, an event is a discrete piece of the entity. In this context, an event is described as an action occurring inside the ecosystem, such as transmitting a message.

(a) key: A identifier for corresponding entries in several databases.

(b) current node: This is the node that holds routing-related messages.

(c) destination node: This is the node where the message should be delivered.

(d) neighbor nodes: A set of all node ids in the same range as the current node.

(e) message-id: the identifier for the message to be sent.

(f) selected_ forwarder: The node chosen as the message's forwarder in the PRoPHET algorithm.

(g) predictability of the current node: Delivery prediction computed by the original PRoPHET algorithm for the current node.

(h) predictability of the selected forwarder: Delivery forecast for the chosen forwarder, based on which it was chosen as the best forwarder.

2. Message Dataset: This dataset comprises rows that define the message's information, such as its source, final destination, message size, and path travelled. It comprises delivery status, which determines if the message was delivered to the final destination or whether it was rejected in the middle of the process. The dataset was further divided into three sections based on this feature:

(a) Delivered: This dataset included messages that were eventually delivered to their intended recipients.

b) Aborted or dropped: Messages that are lost in the routing process.

(c) Relayed: Messages that are conveyed to intermediate nodes.

Messages that are relayed to intermediary nodes are referred to as

As columns of the row, the following information is contained in the characteristics of this dataset:

• key: As previously explained.

• message-id: A one-of-a-kind identifier for the message.

• message size: The message's size in bytes.

• source: The node that generates the message.

• destination: The node where the intended message should be sent.

• number of _hops: The number of nodes passed through on the way to the destination.

• message creation time: In simulated seconds, the time it took to produce the message.

• message deletion time: In simulated seconds, the time the message was removed.

Algorithm 1: Labelling the dataset.

Result: Labelled dataset that can be fed into the model

```
Begin ;
foreach event e in set E do
    foreach message M in set M_s do
        /* if key of the records are same and the
           current_node and selected_forwarder_node are
           present successively in the path of
           themessage and message_id in event e and
           message_id in message M are same
        */
     if message_id[M] == message_id[e] and Key(e) == Key(M)
    _and current_node(e) + selected_forwarder(e) in path[M]
        then
            delivery_status[e] = “deliver";
            processed[e]  =  True;
    _
    foreach event e in set E  do
        if processed[e] != True then
            delivery_status[e] = “not_delivered";
            processed[e] = True
    _  _
    foreach event e in set  E do
        remove processed[e];
    _
```

3.Event meta-data dataset: This dataset describes every entity engaged in every event in the event dataset, such as the source node, current node, intermediate node, and selected forwarder node.

(a) Host info: The node's host info is a collection of data about the node at a specific point in time. This comprises the y and x coordinates of nodes, buffer size, next time to move, free buffer space, and node speed, which are all reported in 9 columns. For each event recorded, the characteristics of this dataset comprise the following information in columns of the produced dataset.

• key: As formerly mentioned.
• hc: Current node's host information.
• hd: Information about the target node's host.
• hn: Host information with all of your neighbours.
• fid: The selected forwarder's identifier.
• message-id: As previously said.

After this stage, sufficient datasets with sufficient rows were created to pre-process the information and train the algorithm.

4.2.2 Stage 2: Building and Training the model

After the datasets have been extracted into a usable format, the event dataset is pre-processed to add an extra feature called delivery status. The suggested method to anticipate nodes while making routing decisions gains an additional element in the form of a label from delivery status. The label is a categorical variable that has two values: delivered and undeliverable. A delivered label for a row in the event dataset indicates that the message conveyed as part of the event was delivered successfully, while not delivered suggests that the message was not delivered successfully. The following section gives a quick overview of how labeling is done.

1.Messages from the message dataset's delivered category were evaluated. The route feature includes a list of every node the message passed through to arrive at its destination. Let's pretend that nodes n1 and n2 with key k were two of the subsequent nodes in the path for the message M. This implies that in some event recorded in the event dataset, a message was sent from node n1 to node n2.

2.Using the above data, locate a row in the event dataset that corresponds to the above data. Find a row that meets the requirements listed below.

• current node = n1
• chosen forwarder = n2
• key = k
• message_id = M

The rows that satisfy the criteria above are labeled as delivered, while the remainder is classified as not delivered.

The final datasets are created using Algorithm 1 from the intermediate datasets collected throughout the experiment. The final dataset is used to train the random forest classifier to determine which nodes are credible and not.

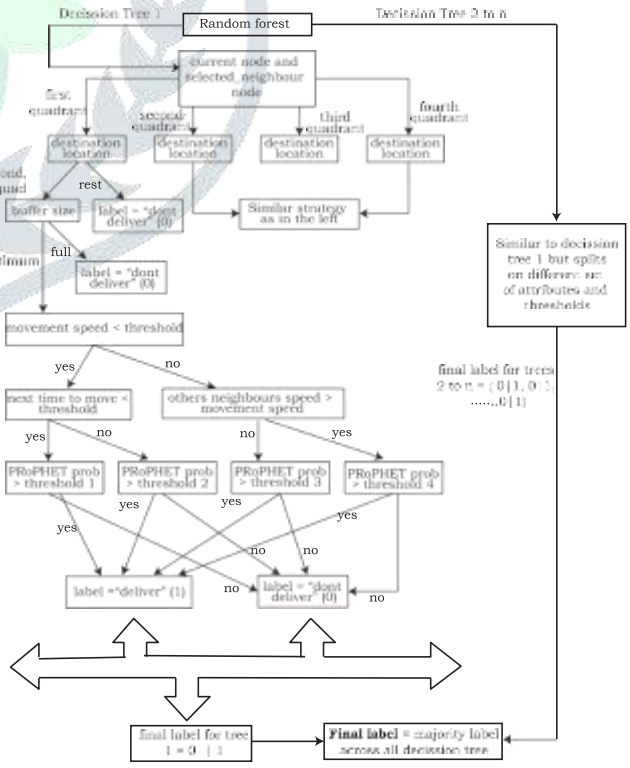

Fig. 1. Random forest architecture.

4.3 Usage of random forest in proposed algorithm

Random forest is used in the suggested approach.

For both classification and regression, Random Forest is one of the most used machine learning methods. The existing PRoPHET algorithm will not use historical data and relies solely on current context information in its static model. As a result, incorporating machine learning capabilities into it might make it more robust. Random Forest uses the ensemble learning approach. Building many trees and aggregating them will only make the model more reliable. In addition, because it employs boosting and bagging techniques, it gives superior classification and helps to minimize overfitting when compared to its competitors. Because random forest can operate with multi-dimensional input data and the data used to train the model has a dimension, it is more suitable for this application. Each node in an opportunistic network environment carries knowledge about its neighbors' context and position. To generate predictions, this contextual information is combined with the suggested model. While making every routing decision, the random forest design is employed to separate reliable from unreliable nodes (see Fig. 1). In the random forest approach, the architecture consists of n number of decision trees, which are the ideal choice of base models/learners. The outputs of these basic models are then combined via bootstrap aggregation, commonly known as bagging, to obtain the final results. The Majority vote is a combining strategy used in classification problems that provide the output/label that the majority of the base learners anticipated. The decision tree for one of the n decision trees used in the random forest design has been extended in this figure. Each base decision tree is trained using a random sample of inputs, replacements, and a new set of features to produce an overall impact of merging the findings, which reduces variance in the final output.

Table 1
Simulation parameters.

Parameter	Value
Simulation Area	(4500 × 3400) sq.m
Total number of nodes	198
Total nodes group	6
Pedestrains group	3
Nodes number in each pedestrian group	30
Pedestrian walking speed	0.5-1.5 km/h
Pedestrian Buffer size	15 Mb
Tram groups	3
Nodes number in each tram group	2
Tram speed	6.5 Km/h
Tram Buffer size	59 Mb
Transmission speed of Bluetooth	250 K
Transmission range of Bluetooth	20 m
High speed interface transmission speed	10 Mb
High speed interface transmission range	1500 m
TTL for message in every group	100 min
Message generation intervals	25–35 s
Size of message	500 Kb - 1 Mb
Mobility model	Shortage path based
movement model	
Runtime of Simulation	100,000 s

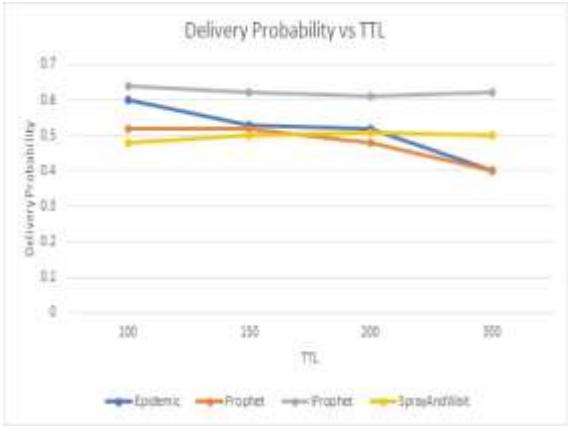

Fig. 2. Comparison of the delivery probability vs TTL.

The first node in decision tree one analyses the current node's position coordinates and the location coordinates of the selected neighbor node. It makes a choice based on the quadrant of the neighbor chosen concerning the present node. The selected neighbor node is the one with which the current node has formed a connection, and the current node must decide on the dependability of the connection before passing it on to the neighbor node. To advance to the next decision node, the next decision node analyses the position coordinates of the target for each quadrant. Suppose the Selected neighbor node is in the first quadrant. In that case, the destinations in the first, second, and fourth quadrants are legitimate. The selected neighbor node may eventually move, allowing the decision tree to progress to the next level.

Similarly, neighbors in other quadrants, i.e., destination node quadrants encircling neighbor quadrants, are deemed legitimate. For example, the surrounding quadrants for the second quadrant are 1,2,3, while the third quadrant is 2,3,4. The message with that destination is not transmitted to the Selected neighbor node if the destination is not in the surrounding quadrant. The buffer size of the Selected neighbor node is taken into account by the following decision node. The nodes with the largest free buffer size to handle the incoming message are advanced down the tree, while the nodes with the smallest free buffer size are eliminated (i.e., the message is not forwarded). The next decision is made based on the Selected neighbor node's movement speed criterion. The movement speed is compared to the threshold computed by the decision tree, and alternative pathways are permitted to be taken. The nodes are not instantly considered unreliable if the movement speed is less than the threshold, as stated in the decision nodes. Instead, the decision nodes will compare the feature value to that of many other neighbors' feature values to make a reliable judgment.

Finally, for final message transmission, the PRoPHET values of selected neighbor nodes that match the threshold established by their respective groups are considered. This method is repeated for additional base learners using a new collection of features and randomly sampled data. Extracted data about one neighbor at a time will be input into the trained model for a specific node. The model categorizes the neighbors as either trustworthy forwarders or non-reliable forwarders based on the attributes. Data

packets are sent to all credible forwarders after traveling through all of the neighbors, as shown in Algorithm 2. Because it is a step forward from the previous version, the Prophet probability is also one of the features for classification in the PRoPHET algorithm. The source code of the PRoPHET algorithm in ONE simulator was modified to include the trained model to examine how the model behaved in the actual world.

Algorithm 2: Intelligent PRoPHET algorithm.

```
1 Begin ;
// Original PRoPHET
2 foreach node Ni as i do
3     dp[i] <- random()
4 for every 5 s do
5     foreach node Ni as i do
6     dp[i] < - GetPRoPHET Probabilty(i);

// End of original PRoPHET
7 foreach message m in current node n buffer do
8     foreach node Ni as n in neighborhood set G do
9     P(n) < - dp[n];
10    if dp[n] ≥ P(Ni ) then
11    if Ni == X then
12    update X;
13    update G;
14    Label = RandomF orest1(X, G, P(X), P(n));
15    if Label == "deliver" then
16    Relay the message to node X ;
17    else
18    if Label == "don't deliver" then
19    Don't relay the message to the node X ;

20    if dp[n] <P(Ni ) the
21    if Ni == X then
22    update X;
23    update G;
24    Label = RandomF orest2(X, G, P(X), P(n));
25    if Label == "deliver" then
26    Relay the message to node X ;
27    else
28    if Label == "don't deliver" then
29    Don't relay the message to the node X ;
```

V. EXPERIMENTAL EVALUATION

This section explains the environment used to simulate the suggested study and a summary of the outcomes achieved.

5.1. Simulation settings

The Opportunistic Network Environment (ONE) [20] simulator is used to evaluate the proposed iPRoPHET in this section. The simulation is run on a 4500 X 3400 sq. m landscape with 51 to 198 nodes distributed randomly. The node's broadcast range is set at 20 meters.

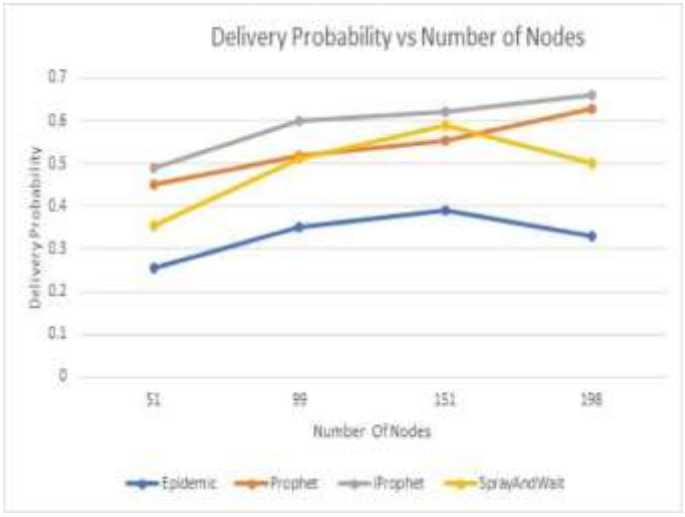

Fig. 3. Comparison of the delivery probability vs no. of nodes.

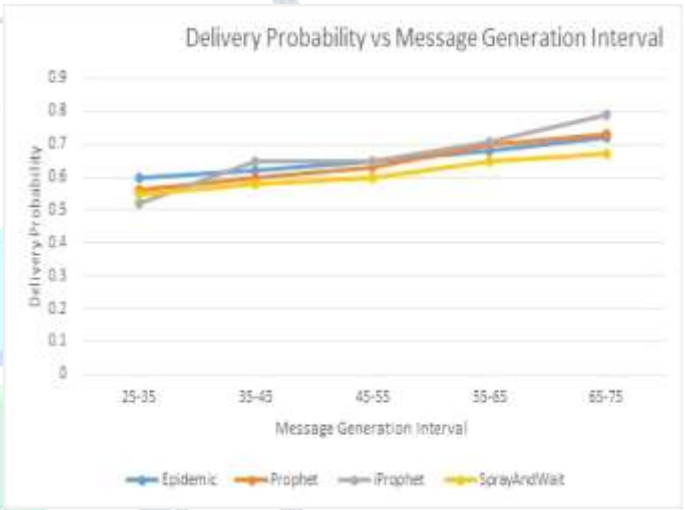

Fig .4 . Comparison of delivery probability vs message generation interval

Table 1 lists the many parameters that were taken into account during the simulation. The proposed iPRoPHET routing protocol was compared against Epidemic [12], Prophet [14], and Spray and Wait [21] routing protocols, with TTL values ranging from 100 to 300 seconds, node counts ranging from 51 to 198, and message generation intervals ranging from 25 to 35 seconds.

5.1. Simulation results and performance analysis

In this section, the proposed algorithm's performance is compared to similar current algorithms in terms of delivery probability, buffer duration, hop count, overhead ratio, and latency.

The delivery probability of several algorithms is compared in Figs. 2–4 when the TTL, Number of Nodes, and Message Generation Period are changed. It can be shown that the delivery probability of algorithms

otherthan Epidemic and iPRoPHET falls as TTL increases because TTL directly increases message lifespan, causing messages to stay in the node buffer for longer than expected, resulting in more messages being lost. The delivery probability rises with the iPRoPHET method because the next forwarder is chosen using a random forest

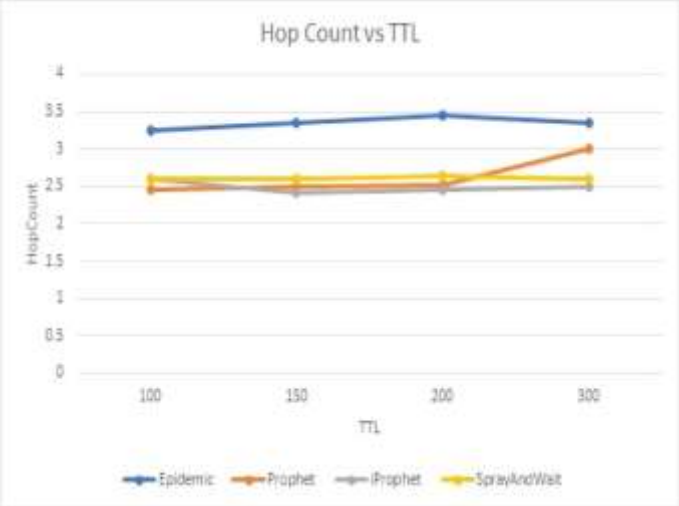

Fig. 5. Comparison of the hop count vs TTL.

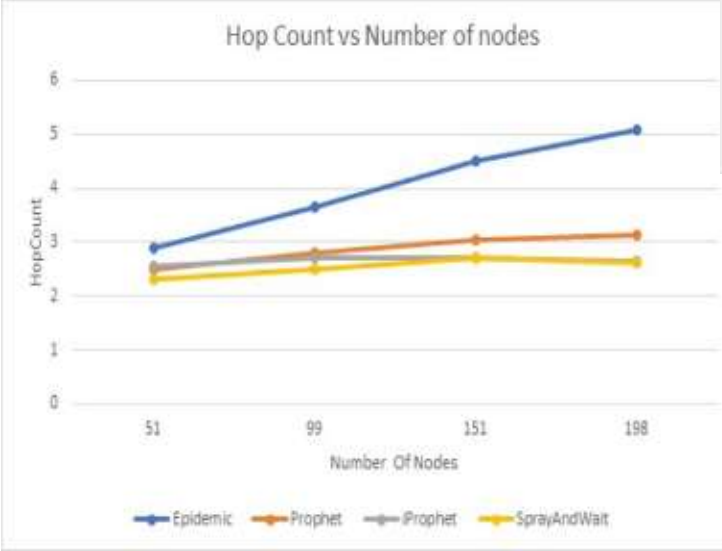

Fig. 6. Comparison of the hop count vs no. of nodes.

classifier that includes PRoPHET probability and other contextual information about the forwarders. The next forwarders are selected not just when their PRoPHET probability is higher but also when it is lower, allowing the messages in the buffer to be cleared as fast as feasible. When the number of nodes is raised, the algorithm can deal with additional forwarders, which improves message delivery accuracy. As the message generation interval is extended, it can be shown that the iPRoPHET algorithm beats all other algorithms in terms of delivery probability.

When the message interval is extended, the network overhead is reduced, and the router queues are free to accept the messages, resulting in fewer messages being dropped from the node's buffer queue. Figure 11 shows how the overhead ratio has decreased (Overhead ratio vs. Message Generation Interval). This, together with the use of suitable forwarders, increases the chances of messages being delivered to their intended recipients. The Hop Count of several algorithms is compared in Figures 5–7 as the TTL, Number of Nodes and Message Generation Interval are changed. It can be observed that the iPRoPHET algorithm has a lower hop count than the PRoPHET method in all variants. This is because the iPRoPHET algorithm seeks to maximize delivery probability by considering forwarders with lower PRoPHET probabilities which may have superior contextual characteristics.

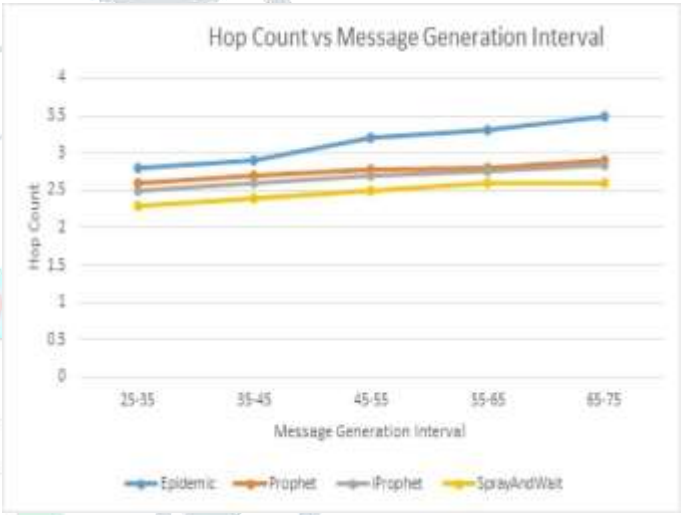

Fig. 7. Comparison of the hop count vs message generation interval.

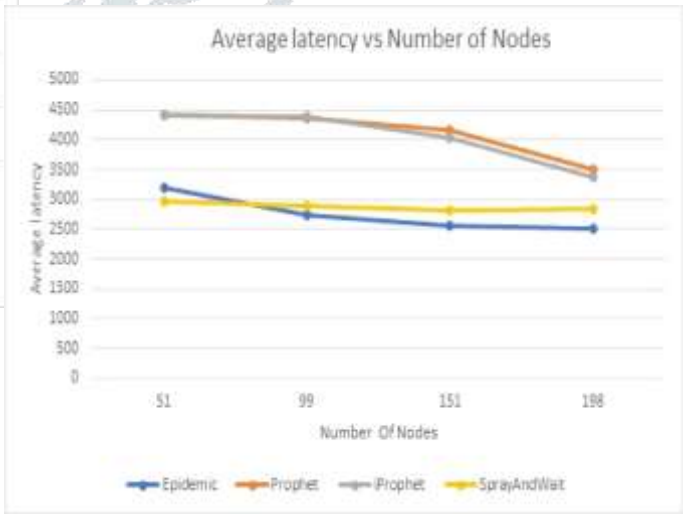

Fig. 8. Comparison of the average latency vs no. of nodes.

When the number of nodes is raised, the number of forwarders available to route messages increases in proportion. The random forest classifier chooses the fit nodes appropriately among these forwarders, resulting in a lower overall hop count. When compared to TTL, the same reasoning applies. Because of their nature of

selecting forwarder nodes that are less likely to forward it to destination, the hop count of other algorithms except SprayAndWait grows as TTL increases.

Furthermore, with PRoPHET, messages stay in the buffer for extended periods of time because nodes will not forward messages if the PRoPHET probability for the forwarders is low. The amount of messages in the network decreases when the message production interval is changed, while the classifier's accuracy remains constant. As a result, the general drop in hop count can be seen in all of the graphs.

The average delay of several methods is compared in Figures 8–10 when the number of nodes, TTL, and Message Generation Interval are altered. The iPRoPHET algorithm has a slightly shorter average latency time than PRoPHET because it selects nodes with lower PRoPHET delivery probability, which is needed to maximize the overall delivery probability.

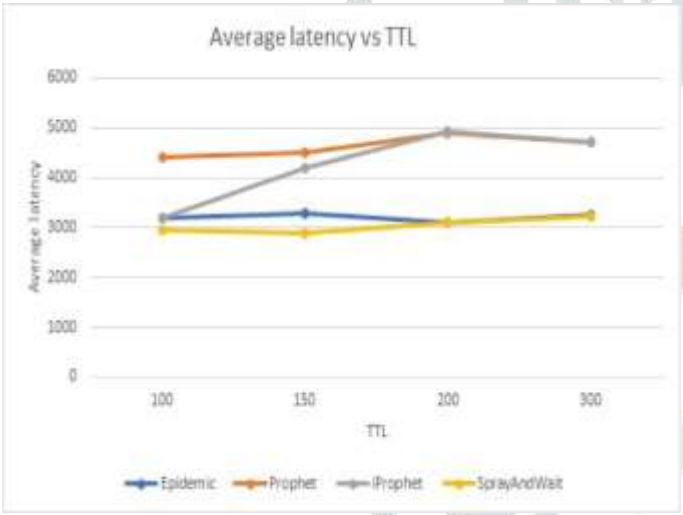

Fig. 9. Comparison of the average latency vs TTL.

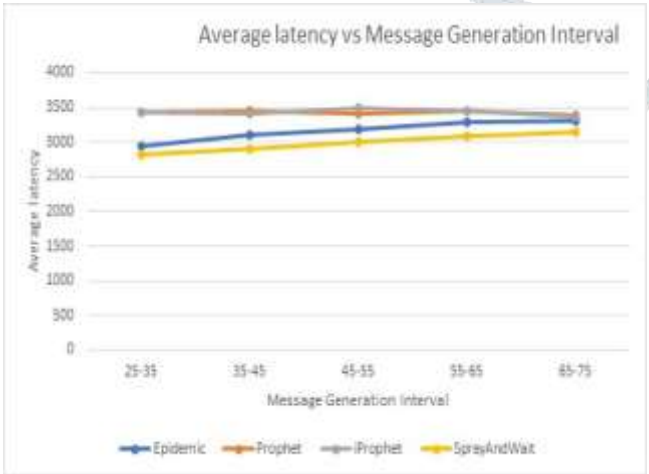

Fig. 10. Comparison of the Average latency vs message generation interval.

. However, in doing so, the algorithm could choose a few extremes that may keep messages for far longer than standard, and as the number of nodes increases, the outlier count increases, which increases the overall delivery probability. However, this is offset by the non-outlier nodes' lower hop count delivery, which helps keep latency periods under control for most messages. When the TTL and message production interval is changed, outliers have a similar impact. However, the majority of non-outlier nodes overcome this effect by delivering messages to their destinations on time. The overhead ratio of several methods is compared in Figure 11 for message production intervals ranging from 25 to 35 to 65–75. Because of the higher delivery probability of stronger forwarders selected as part of the random forest classifier, the overhead ratio in aPRoPHET is better.

The average buffer time of several algorithms is compared in Figures 12 and 13 when the number of nodes and TTL are changed. The aPRoPHET algorithm has a slightly longer average buffer time due to the algorithm's selection of nodes with lower PRoPHET delivery probability, which is required to maximize the overall delivery probability. However, in doing so, the algorithm may select a few outliers who may keep messages for far longer than usual, and as the number of nodes grows, the outlier will become more prominent. When the TTL is raised, the impact of outliers is seen to be identical. The messages remain in the outlier nodes' buffer for longer than normal before being forwarded, resulting in a longer average buffer duration.

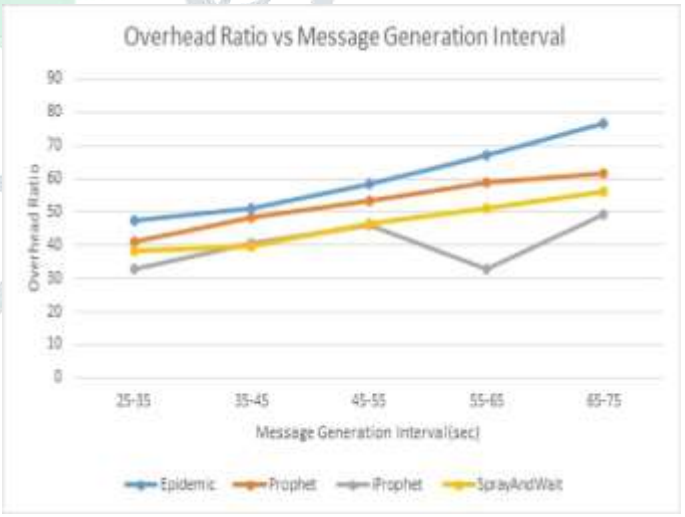

Fig. 11. Comparison of the overhead ratio vs message generation interval

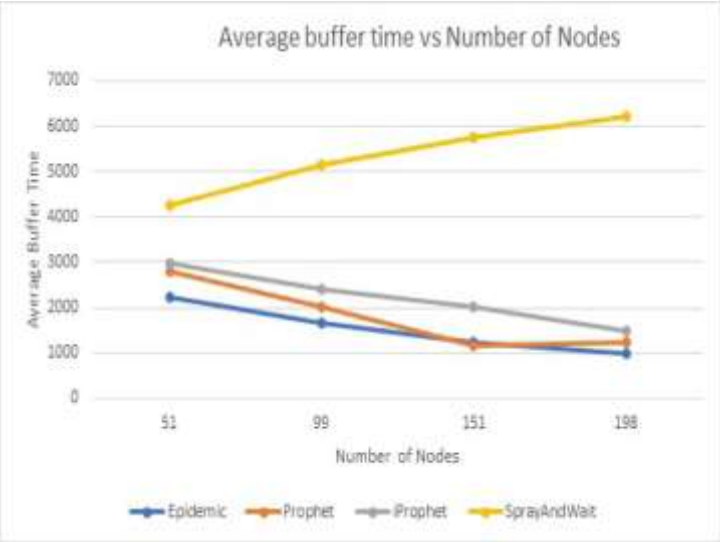

Fig. 12. Comparison of the average buffer time vs no. of nodes.

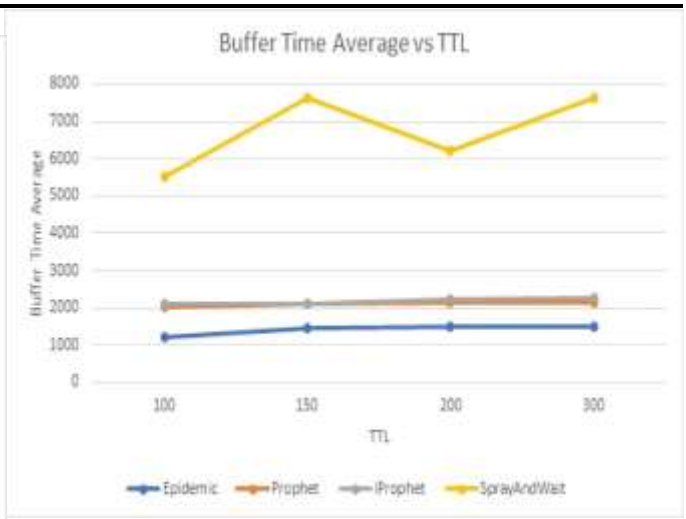

Fig. 13. Comparison of the buffer time average vs ttl.

VI .CONCLUSION

With the introduction of opportunistic mobile networks as a new mechanism for wireless transmissions, communications in opportunistic mobile networks are based on the creation of opportunistic connections among mobile nodes rather than the availability of end-to-end communication channels. However, with opportunistic IoT, data gathering, propagation, and distribution require opportunistic connections that use mobility, short-range transmission, and technology.

Routing protocol configuration presents a difficulty during such operations owing to its inherent complexity in establishing connectivity among devices and identifying an appropriate forwarder to forward data packets to their target destination. As a result, the iPRoPHET (intelligent PRoPHET) multi-copy routing method for opportunistic networks is introduced in this study. To establish message routing, this technique employs a machine learning classifier known as random forest and the delivery probability of existing PRoPHET. Based on the simulation findings, the following conclusions may be drawn :(i) Random forest classifies nodes as dependable or unreliable while making routing decisions, resulting in a better selection of intermediate nodes for efficient routing. (ii) In terms of delivery probability, hop count, overhead ratio, and latency, the proposed technique outperforms a comparison to current algorithms of a similar nature

(iii) The classifier is trained with a significant quantity of data retrieved through simulation, which results in accurate classification of nodes as reliable or unreliable, allowing for efficient communication. However, compared to similar multi-copy routing algorithms, the suggested approach would perform poorly in terms of average buffer time. Due to the presence of outlier nodes, the average buffer duration has encountered a performance issue, which has to be addressed as part of future development. Furthermore, in future development, the nodes' energy consumption must be improved to operate with compute-intensive machine learning methods.

VI. REFERENCES

[1] B. Guo, D. Zhang, Z. Wang, X.Z. Z. Yu, Opportunistic IoT: exploring the harmonious interaction between human and the internet of things, J. Netw. Comput. Appl. (2013) 1531–1539.

[2] L. Pelusi, A. Passarella, M. Conti, Opportunistic networking: data forwarding in disconnected mobile ad hoc networks, IEEE Commun. Mag. 44 (11) (Nov. 2006) 134–141.

[3] E. Spaho, L. Barolli, V. Kolici, A. Lala, Evaluation of single-copy and multiple-copy routing protocols in a realistic VDTN scenario, in: 2016 10th Inter- national Conference on Complex, Intelligent, and Software Intensive Systems (CISIS), IEEE, 2016, pp. 284–289.

[4] A. Keranen, J. Ott, T. Karkkainen, The ONE simulator for DTN protocol evaluation, in: Proceedings of the 2nd International Conference on Simulation Tools and Techniques, ICST (Institute for Computer Sciences, Social-Informatics and Q, 2009, p. 55.

[5] A.S. Sohal, R. Sandhu, S.K. Sood, V. Chang, A cybersecurity framework to identify malicious edge device in fog computing and cloud-of-things environ- ments, Comput. Secur. 74 (2018) 340–354.

[6] R. Amin, N. Kumar, G. Biswas, R. Iqbal, V. Chang, A light weight authentication protocol for IoT-enabled devices in distributed cloud computing environment, Future Gener. Comput. Syst. 78 (2018) 1005–1019.

[7] G. Sun, V. Chang, M. Ramachandran, Z. Sun, G. Li, H. Yu, D. Liao, Efficient location privacy algorithm for internet of things (IoT) services and applica- tions, J. Netw. Comput. Appl. 89 (2017) 3–13.

[8] G. Sun, S. Cai, H. Yu, S. Maharjan, V. Chang, X. Du, M. Guizani, Location privacy preservation for mobile users in location-based services, IEEE Access 7 (2019) 87425–87438.

[9] A.Vahdat, D.Becker, Epidemic Routing for Partially Connected ad hoc networks, TechnicalReportCS-2000-06, 2000.

[10] S.K. Dhurandher, D.K. Sharma, I. Woungang, S. Bhati, HBPR: history based prediction for routing in infrastructure-less opportunistic networks, IEEE AINA 2013, Barcelona, Spain,March 25–28,, 2013.

[11] A. Lindgren, A. Doria, O. Schelen, Probabilistic routing in intermittently connected networks, ACM SIGMOBILE Mob. Comput. Commun.Rev. 7 (3) (July 2003) 19–20.

[12] S.K. Dhurandher, S.J. Borah, M.S. Obaidat, D.K. Sharma, B.B. S. Gupta, Probability-based controlled flooding in opportunistic networks, in: Proc. of Intl. Conference on Wireless Information Networks and System (WINSYS), Colmar, France, July 20–22,, 2015, pp. 3–8.

[13] C. Boldrini, M. Conti, I. Iacopini, A. Passarella, HiBOp: a history based routing protocol for opportunistic networks, in: Proc. of IEEE Intl. Symposium on World of Wireless, Mobile and Multimedia Networks (WOWMOM 2007), 2007, pp. 1–12.

[14] S.K. Dhurandher, S. Borah, I. Woungang, D.K. Sharma, K. Arora, D. Agarwal, EDR: an encounter and distance based routing protocol for opportunistic networks, in: Proc. of IEEE AINA 2016, Crans-Montana, Switzerland,March 23–25,, 2016, pp. 297–302.

[15] I. k Jeon, K. w. Lee, A dynamic Markov chain prediction model for delay-tolerant networks, Int. J. Distrib. Sens. Netw. 12 (Sept. 2016) 2–7.

[16] D.K. Sharma, S.K. Dhurandher, I. Woungang, R.K. Srivastava, A. Mohananey, J.J. Rodrigues, A machine learning based protocol for efficient routing in opportunistic networks, IEEE Systems Journal (2016). doi:10.1109/JSYST.2016.2630923

[17] A. Keranen, Opportunistic network environment simulator, Special Assignment report, Helsinki University of Technology, Department of Communica- tions and Networking, 2008.

[18] T. Spyropoulos, K. Psounis, C.S. Raghavendra, Spray and wait: an efficient routing scheme for intermittently connected mobile networks, in: Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking, ACM, 2005, pp. 252–259.