CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Capacity and contention-based joint routing and gateway selection for machine-type communications
Authors	Farooq, Muhammad O.;Sreenan, Cormac J.;Brown, Kenneth N.
Publication date	2017-04-20
Original Citation	Farooq, M. O., Sreenan, C. J. and Brown, K. N. (2017) 'Capacity and contention-based joint routing and gateway selection for machine-type communications', Ad Hoc Networks, 62(Supplement C), pp. 35-49. doi:10.1016/j.adhoc.2017.04.006
Type of publication	Article (peer-reviewed)
Link to publisher's version	10.1016/j.adhoc.2017.04.006
Rights	© 2017, Elsevier B.V. This manuscript version is made available under the CC-BY-NC-ND 4.0 license. - http://creativecommons.org/licenses/by-nc-nd/4.0/
Download date	2026-11-16 19:26:35
Item downloaded from	https://hdl.handle.net/10468/5054



UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Capacity and Contention-based Joint Routing and Gateway Selection for Machine-Type Communications

Muhammad Omer Farooq, Cormac J. Sreenan,
Kenneth N. Brown
CTVR, Department of Computer Science, University College Cork, Ireland
Email: omer.farooq@insight-centre.org, cjs@cs.ucc.ie, k.brown@cs.ucc.ie

Abstract

Typically, in machine-type communications (MTC) devices communicate with servers over the Internet. In a large-scale machine-to-machine area (M2M) network, the devices may not connect directly to the Internet due to radio transmission and energy limitations. Therefore, the devices collaborate wirelessly to relay their data to a gateway. A large-scale M2M area network may have multiple gateways, selecting a proper gateway for the devices can have immense impact on the network's performance. We present the channel capacity and contention-based joint routing and gateway selection methods for MTC. Based on channel capacity and contention, our methods select the best gateway on per-packet, per-flow, and per-node basis. We compare the methods' performance with existing methods using simulation and test-bed experiments. We analyse the impact of the number of gateways, physical distribution of transmitters, control overhead, and duty-cycling on the performance of the gateway selection methods. Our results demonstrate that, in duty-cycled operations, the methods' performance depends on control overhead and making a good trade-off between load imbalance to different gateways and a forwarding path's length. Otherwise only the latter impacts the methods' performance. In general, our node-based best gateway selection method makes a better trade-off and exhibits lower control overhead, hence it demonstrates better performance. Moreover, our methods demonstrate better performance as compared to an existing state-of-the-art joint routing and gateway selection method.

Keywords. Machine-Type Communications, Routing, Gateway Selection, Channel Capacity, Contention

1 Introduction

Machine-type communication (MTC) is a form of data communication that allows smart networked devices to communicate with each other without human intervention [1]. It is an essential component for many smart city applications including smart roads and intelligent transportation, smart homes, smart parking, and waste management [2]. It is anticipated that MTC will have significant economic impact, and the total market size by 2025 is estimated to be 30 billion connected devices, with 7 billion MTC devices connected to the Internet through cellular networks [3].

Figure 1 shows a general communication architecture of MTC. We elaborate the architecture with the help of a smart roads and intelligent transportation system. In the system, many smart devices monitor traffic on a network of roads, and may report accidents, traffic rule violations,

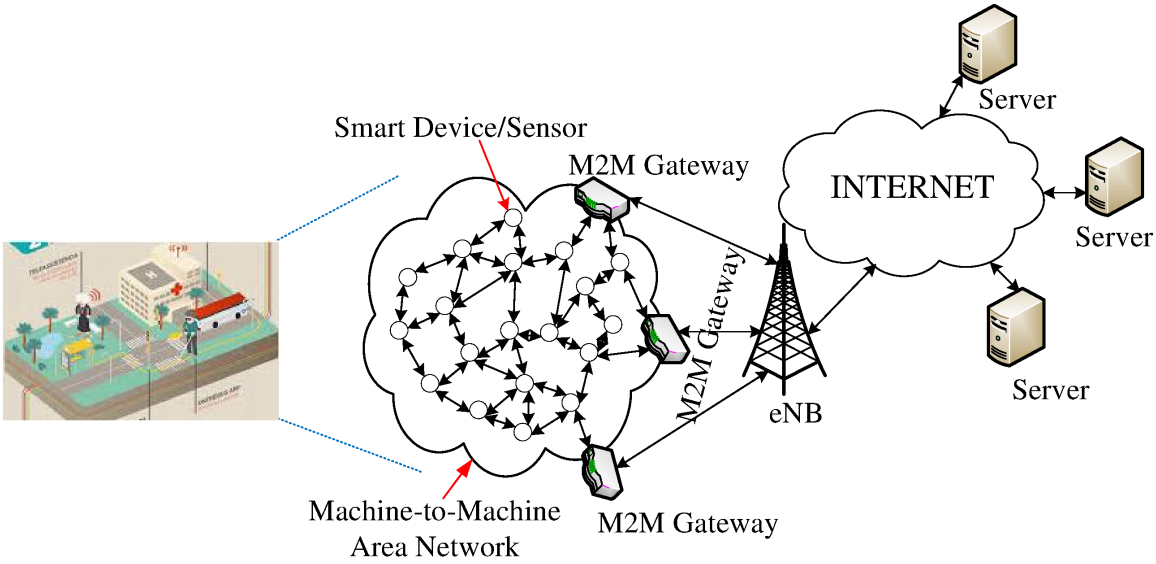


Figure 1: A Communication Architecture for Machine-Type Communication

or traffic load on different roads. Such a network can span a large geographical area, therefore it is possible that due to the power and radio transmission range limitations, the devices may not be able to communicate directly with a base-station. Hence, the devices in machine-to-machine (M2M) area network wirelessly collaborate in an ad-hoc manner to relay the data to a M2M gateway. The M2M gateway, uses an access network, for example, a cellular network (future 5G cellular networks are envisioned as preferred access networks for MTC because of their ubiquitous nature and enhanced data rates [4]) to connect to the Internet in order to forward the data to any one or multiple MTC servers. Based on the received data the servers may alert an appropriate entity, for example, in case of a congestion on roads, the servers feed data to a global position system (GPS), and the GPS may then change different vehicles' routes.

In case of a single M2M gateway in a M2M area network, all data packets merge near the gateway. Hence, the probability of congestion is higher near the gateway. The congestion can negatively impact a network's performance, therefore typically multiple gateways are deployed in such networks to enhance the performance. It is importance for a smart device to select the best gateway to relay its packets, as otherwise the anticipated performance gains through the deployment of multiple gateway may not be achieved. An effective gateway selection method can improve the network throughput and reliability. On the contrary, a poor selection method can relay most of the devices' packets to the same gateway, thus negatively impacts throughput, delay, and reliability due to the congestion. Typically, the devices in such networks are battery operated, therefore it can also reduce the network's lifetime as the devices near the gateway deplete their energy faster.

As well as the gateway selection method, the routing metric to select data forwarding paths also has an impact on the choice of gateway. If the shortest hop-count metric is used and each device selects the closest gateway, it is possible that the selected path and gateway are already congested. Similarly, if a protocol discovers routes to different gateways using a metric that considers data load on different paths and each device selects the gateway to which it has the least loaded path, the network performance may improve. But, in this case, the device may select a far away gateway, and the accumulated time required to access the channel on the

longer path along with the accumulated required transmission power may offset or negatively impact the advantage of selecting the least loaded gateway. Therefore, we present joint routing and gateway selection methods for MTC with an aim of balancing forwarding path length and channel capacity. Our routing and gateway selection methods use minimum end-to-end channel capacity and contention to select forwarding path and gateway. Our main contributions are as follows:

i. Channel capacity and contention-based joint routing and gateway selection methods for MTC.

ii. An exhaustive evaluation of the limits of per-packet, per-flow, and node-based gateway selection methods.

iii. Analysing the impact of the number of gateways, duty-cycling, physical distribution of transmitters, and control overhead on the selection methods.

iv. Our static network topology simulation results and our test-bed results that demonstrate that our per-packet, per-flow, and per-node based best gateway selection methods are up to 325%, 67%, and 33% better in load-balancing to different gateways respectively as compared to the closest gateway selection method. Our per-packet and per-node best gateway selection methods demonstrate up to 22% and 13% higher packet delivery ratio respectively as compared to the closest gateway selection. Moreover, our methods demonstrate up to 37% lower per-packet end-to-end delay and up to 48% lower total number of retransmissions as compared to a state-of-the-art available-bandwidth-based method. Furthermore, the results reveal that making a good trade-off between load imbalance to different gateways and path length is an important property of a good joint routing and gateway selection protocol. As per the results our node-based best selection method better makes use of this property as compared to the other protocols.

The remainder of this paper is organized as follows. Section 2 presents related work. Our different channel capacity and contention-based joint routing and gateway selection protocols are presented in Section 3. Simulation and test-bed results are presented in Section 4, and conclusions are presented in Section 5.

2 Related Work

Categorization of joint routing and gateway selection methods for wireless networks along with example existing protocols in each category are shown in Figure 2. In this section, we present a discussion on different protocols in each category. In the rest of this paper, we use the terms protocol and method interchangeable.

2.1 Dynamic Gateway Selection Methods

Depending on a metric for a gateway selection, at a source node dynamic methods assign the best gateway to each data packet. Such methods continuously monitor data forwarding paths to different gateways and/or the status of the gateways. As soon as a better gateway becomes available, the methods switch the gateway. Afterwards, a routing protocol relays data packets originating from the node to the newly selected gateway. Such methods can be categorized as single-metric-based and composite-metric-based methods.

2.1.1 Single-Metric-Based Methods

The methods presented in [5–8] use energy consumption as the metric to select the best gateway. In [5] a mobile anchor is used to monitor energy consumption of nodes near different gateways, depending upon the residual energy of the nodes, the mobile anchor partitions a network into different zones to balance the energy consumption. In [6] and [7] energy depletion rate of nodes is used as an indication of the data traffic load on nodes. A node selects the gateway to which it has the path with the lowest energy depletion rate. In [8], the shortest hop-count is used to select the best gateway, but the hop-count value is adjusted based on nodes' energy consumption along a path, i.e., if the energy consumption on the path increases the method increases the hop-count distance to the gateway in an attempt to direct traffic to other less loaded gateways. The methods presented in [9] and [10] try to maximize the network throughput by using the throughput metric, i.e., a node selects the gateway to which it may achieve the highest throughput.

2.1.2 Composite-Metric-Based Methods

Using a single metric, for example, energy consumption or throughput can result in a poor gateway selection. A single-metric-based method can select a gateway that is far away, hence the cost of relaying packets to the best gateway may offset or even negatively impact the benefit of selecting the best gateway. This is highly likely in wireless networks because of shared and error-prone communication links. For gateway selection in [11], a composite metric is formed by combing traffic load on a forwarding path to a gateway, hop-count to a gateway, and quality of links on the path to a gateway. The goal is to enhance network capacity, fairness, and reliability. In [12], hop-count to a gateway and the traffic load on the gateway is combined to select the best gateway. The protocol presented in [13], combines expected transmission time (ETT) on different paths to different gateways, and ETT at different gateways to select a gateway. The gateway corresponding to the lowest value of the composite metric is selected. Similarly, [14–16] use combination of hop-count, energy level, interference on candidate forwarding paths, and buffer level at different gateways to select the best gateway. In [17], service distance metric composed of the number of devices connected to a gateway, normalized received signal strength, gateway bandwidth consumed by connected devices, and bandwidth requested by a device is used to select the best gateway. A device sends an attachment request to available gateways, and the gateways forward the request to a centralized server. The centralized server evaluates the suitability of the available gateways, and assigns the device to the gateway with the highest value of the metric. The method proposed in [18] uses a fuzzy algorithm for gateway selection. The goals are to reduce energy consumption and end-to-end delay. The inputs to the fuzzy algorithm are: (i) candidates; number of downstream nodes a node has towards a gateway, (ii) neighbours; number of nodes that connect a candidate downstream node to a gateway, and (iii) percentage of remaining energy at downstream node. The best gateway is selected based on the algorithm's output.

2.2 Flow-Level Fixed Gateway Selection Methods

Depending on a metric for a gateway selection, flow-level fixed gateway selection methods select the best gateway for a flow at the flow's start time. In case of multiple such gateways, a gateway is selected randomly. The selected gateway can change if at-least anyone of the following conditions is satisfied: a flow terminates, a gateway malfunctions, or the path to a gateway breaks. Such methods can also be categorized as single-metric-based and composite-metric-based methods.

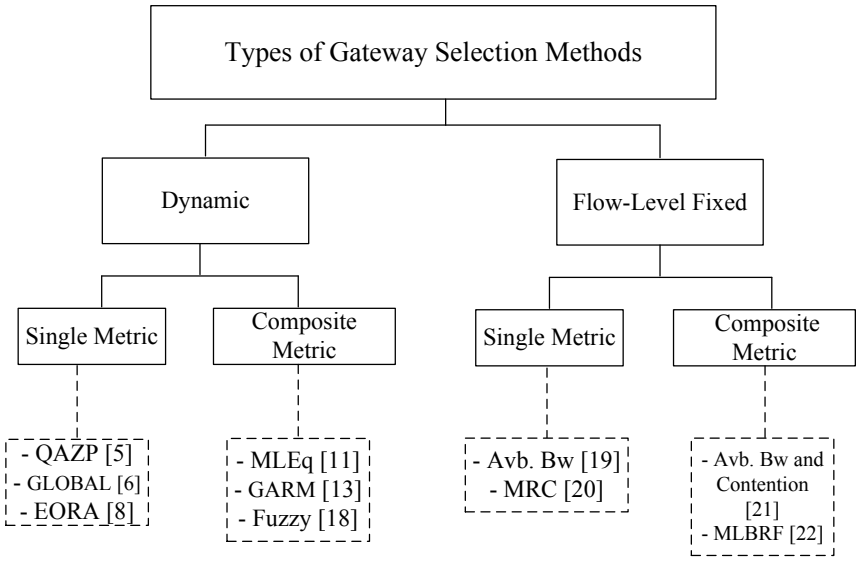


Figure 2: Categorization of Joint Routing and Gateway Selection Methods

2.2.1 Single-Metric-Based Methods

The method presented in [19] selects the best gateway for a flow based on the end-to-end available bandwidth on paths to different available gateways. Therefore, a node selects a gateway for the flow to which it has a path with the highest available bandwidth. Similarly, [20] uses gateways' maximum residual capacity metric, therefore the gateway with the maximum capacity is selected.

2.2.2 Composite-Metric-Based Methods

In [21] a composite-metric-based method is used to select a gateway. The method combines the available bandwidth and contention, i.e., each node in a network measures the impact of contention on its available bandwidth. Based on the metric the routing protocol discovers the best paths to different gateways. At a flow start time, a node selects the gateway towards which it has the best path. In [22] a fuzzy algorithm is used for selecting the best gateway. Different paths to different available gateways are analysed, and the best gateway is selected. The inputs to the algorithm at each node are: contention on links shared with candidate downstream nodes, buffer occupancy at downstream nodes, and geographical distance of downstream nodes from a gateway.

2.3 Discussion

The state-of-the-art gateway selection methods do not explore an option of permanently assigning a gateway to a node at run-time, i.e., assigning and fixing the best forwarding path and gateway for the node at the time the node is about to transmit its first data packet. This approach may help to balance data traffic load to multiple gateways, and it may also avoid longer data forwarding paths. Moreover, less frequent broadcasting of control messages due to the fixing of path and gateway can result in lower control overhead. Therefore, designing and investigating the performance of a node-level fixed gateway selection is an open area for research. Hence, in this paper, we present different channel capacity and contention-based gateway selection methods

and compare per-packet, flow-level fixed, and node-level fixed methods. Moreover, existing work does not consider the impact of physical distribution of transmitters, duty cycling, and control overhead on the methods. Therefore, in this paper, we also analyse the impact of the mentioned factors on not only our designed protocols, but also on random and closest gateway selection methods.

3 Joint Routing and Gateway Selection Protocols

In this section, we present our different channel capacity and contention-based joint routing and gateway selection protocols. We categorize our protocols in three different categories: per-packet, flow-level fixed, and node-level fixed. It is worth mentioning here that, we are the first to explore the node-level fixed category. In each category, our protocols select the best gateway in terms of the end-to-end channel capacity, and contention on a path to a gateway. Only considering the capacity may result in selecting a lengthy path, i.e., a less frequently used longer path may advertise better end-to-end capacity, but the path can result in higher delay and energy consumption along with low packet delivery ratio (PDR). Therefore, to trade-off the capacity and path length we use the capacity and contention on a path as the composite metric for gateway selection.

3.1 Channel Capacity and Contention Estimation

Typically, machines use low-power communication standards, therefore our protocols assume that the IEEE 802.15.4 standard in ad-hoc mode is used by different machines. But, our channel capacity estimation algorithm is generic, and it can be applied to other communication standards, for example, IEEE 802.11. A node estimates its capacity by monitoring the time required to successfully transmit a data frame, i.e., the difference between the time when the frame was successfully transmitted and the time when the frame was available at the head of MAC layer queue. This includes the time required to access the channel through contention, and the time incurred in retransmissions (if any). A node uses Equation 1 to estimate the capacity. In the equation, $Frame_{size}$ represents the size of the frame in bits, T_{tx} is the time when the frame was successfully transmitted, and T_{qh} is the time when the frame became available at the head of MAC layer queue. It should be noted here that, T_{tx} is captured when the MAC layer acknowledgement for the transmitted frame is received.

$$Capacity_{curr} = \frac{Frame_{size}}{(T_{tx} - T_{qh})} \, bps \tag{1}$$

The capacity estimate obtained through a recent successful transmission can be misleading due to the nature of wireless channel, i.e., contention, reflection, refraction, diffraction, and multi-path fading. Therefore, we use the exponentially weighted moving average method to obtain the estimate, as shown in Equation 2. In the equation, $Node_{cap}$ is a node's capacity, $Capacity_{avg}$ is the previous value of $Node_{cap}$, and α is the smoothing factor. To better reflect the current channel conditions on the capacity, we use $\alpha = 0.33$ [15]. To estimate a node's capacity based on the first successful data frame transmission, $Capacity_{avg}$ is set to the maximum data rate supported by a communication standard, and in our implementation we use the typical data rate supported by the IEEE 802.15.4 standard, i.e., 250 kbps.

$$Node_{cap} = (\alpha \times Capacity_{avg} + (1 - \alpha) \times Capacity_{curr}) \, bps \tag{2}$$

Our channel capacity estimation algorithm indirectly takes into account the impact of contention caused by nodes within the interference range of the node estimating the capacity. Moreover, the algorithm is able to consider the impact of radio duty-cycling on the capacity. If after transmitting a data packet a radio duty-cycling algorithm executing on a node switches off the node's radio, it takes longer time to successfully transmit the packet at the head of the queue. As per our algorithm, the higher the time duration to successfully transmit the packet the lower the capacity. Using $Node_{cap}$ as a routing metric may result in longer data forwarding paths. The selection of a longer path with more capacity may negatively impact the performance due to the accumulated time required to access the channel and total required transmission power along the path. Therefore, to limit a forwarding path's length, our algorithm trades-off capacity and path length, and this is one of the novel feature of our algorithm. For the trade-off, we use approximated intra-flow contention count. Due to the shared nature of the wireless communication medium, the intra-flow contention is created by relaying nodes along a forwarding path. The bandwidth required by a flow by only considering the intra-flow contention is equal to the intra-flow contention count $\times$ the bandwidth required by the flow. It has been demonstrated in [23] that the maximum intra-flow contention count on a node is 5, and it depends on the node's hop-count distance from the source and destination nodes. In a proactive routing protocol, it is hard for any node to predict the source node whose flow's will traverse the node. Therefore, we approximate the count using the node's hop-count distance to a gateway and the maximum intra-flow contention count. Let g be a gateway, n be any arbitrary node, and $dist_{n \rightarrow g}$ is the hop-count distance to g from n. The approximate contention count is denoted by AC, and $AC = min(dist_{n \rightarrow g}, 5)$. Our joint routing and gateway selection methods divides $Node_{cap}$ with AC to trade-off path capacity and length.

3.2 Gateway Selection

Here, we describe our three different gateway selection methods, namely: per-packet, flow-level fixed, and node-level fixed. In all the methods, the gateway selection is performed at the source of a data packet. We opted for the source-based gateway selection because otherwise if a routing update does not reach to a node in a timely manner, short-lived forwarding loops may occur. Our methods assume that all gateways are equivalent. The nodes on the path to the gateway relay the packet based on the gateway's address present in the packet header, and the forwarding information stored in G_{Table}. G_{Table} is maintained by the routing process, and a record in the G_{table} stores the following information: gateway address, gateway sequence number, hop-count, downstream node's network layer address, minimum capacity on a path to the gateway, minimum capacity to the gateway with contention, and a record time out value.

3.2.1 Per-Packet Gateway Selection

Whenever a packet is generated at a source node, the source node consults its G_{Table}, and selects the best gateway among the discovered gateways, i.e., the gateway to which the node has the highest channel capacity with the impact of contention. Transmitting packets to a particular gateway results in a lower capacity to the gateway, hence in this method, subsequent packets may go to different gateways. This may result in a better load balancing. But, packets belonging to the same flow can go to different gateways, hence this method may not be suitable for protocols/applications that require in-order delivery of packets at a gateway. Hereafter, we refer this method as per-packet best gateway selection (PPBS). Algorithm 1 summaries the gateway selection and packet forwarding for PPBS. In the algorithm, *node_addr* represents the network layer address of the node executing the algorithm, and the purpose of

Algorithm 1: Per-Packet Gateway Selection and Packet Forwarding

```
selected_gateway = Null;
downstream_node = Null;
if data_pkt.src_addr == node_addr then
    selected_gateway = select_best_gateway();
    data_pkt.gateway_addr = selected_gateway;
    downstream_node = get_downstream_node_from_G_Table(selected_gateway);
    relay_pkt(data_pkt, downstream_node);
end
else
    if data_pkt.gateway_addr ≠ node_addr then
        downstream_node = get_downstream_node_from_G_Table(data_pkt.gateway_addr);
        relay_pkt(data_pkt, downstream_node);
    end
end
```

$get_downstream_node_from_G_{Table}(selected_gateway)$ is to get the network layer address of the downstream node to the selected gateway from G_{Table}.

3.2.2 Flow-Level Fixed Gateway Selection

Whenever a new flow emerges at a source node, the source node consults its G_{Table}, and selects the best gateway among the discovered gateways, i.e., the gateway to which the node has the highest channel capacity with the impact of contention. Once a gateway is selected for the flow, all data packets of the flow are relayed to the same gateway. The selected gateway for the flow can only change in case of a route failure. To distinguish between different flows, a source node maintains a flow table (F_{Table}). A record in F_{Table} maintains the following information: port number being used by a flow, selected gateway address, and a record time out interval. Whenever a forwarding module at the source receives a data packet, it executes the following logic. If the source address in the packet header is same as the node's own address, the forwarding module at the node consults F_{Table}. If the port number in the packet header matches any port number in F_{Table}, the forwarding module copies the gateway address from F_{Table} to gateway address in the packet header. If the port number in the header does not match with any port number in F_{Table}, forwarding module selects the best gateway for the flow and inserts a new record in F_{Table}. If the forwarding module does not receive a data packet corresponding to a flow before the time out interval, the corresponding record is deleted from F_{Table}. The motivation for the flow-level fixed gateway selection is to determine the impact of fixing a gateway at flow-level on a network's performance. Hereafter, we refer this method as flow-based best gateway selection (FBS). Algorithm 2 summarizes the gateway selection and packet forwarding for FBS.

3.2.3 Node-Level Fixed Gateway Selection

In node-level fixed gateway selection method, a source node selects a gateway when the node generates its first data packet. The gateway to which the node has the highest channel capacity with the impact of contention is selected. All the data packets originating from the node are relayed to the same gateway. The selected gateway for the node can only change in case of a route failure. When the forwarding module at a node receives a data packet, it compares the source address in the packet header with the node's address. If both addresses match, the module

Algorithm 2: Flow-Level Fixed Gateway Selection and Packet Forwarding

```
1  flow_record = Null;
2  downstream_node = Null;
3  selected_gateway = Null;
4  if data_pkt.src_addr == node_addr then
5  |  flow_record = get_record_from_F_Table(data_pkt.port_no);
6  |  if flow_record ≠ Null then
7  |  |  flow_record.refresh_time_out();
8  |  |  data_pkt.gateway_addr = flow_record.gateway_addr;
9  |  |  downstream_node =
   |  |    get_downstream_node_from_G_Table(flow_record.gateway_addr);
10 |  |  relay_pkt(data_pkt, downstream_node);
11 |  end
12 |  else
13 |  |  selected_gateway = select_best_gateway();
14 |  |  flow_record = add_record_in_F_Table(data_pkt.port_no, selected_gateway);
15 |  |  flow_record.set_time_out(timeout_value);
16 |  |  data_pkt.gateway_addr = selected_gateway;
17 |  |  downstream_node = get_downstream_node_from_G_Table(selected_gateway);
18 |  |  relay_pkt(data_pkt, downstream_node);
19 |  end
20 end
21 else
22 |  if data_pkt.gateway_addr ≠ node_addr then
23 |  |  downstream_node = get_downstream_node_from_G_Table(data_pkt.gateway_addr);
24 |  |  relay_pkt(data_pkt, downstream_node);
25 |  end
26 end
```

checks whether the gateway is already selected for the node. If the gateway is selected, the module copies the selected gateway address to the packet header. Otherwise, the module selects the gateway and copy the gateway address in the packet header. In this method, a node selects the best gateway at the time when the first data packet originates from the node, and afterwards the gateway does not change. Therefore, the gateway HELLO and node HELLO message are only broadcasted for a relatively smaller interval of time. Hence, not only the gateway is fixed, but the forwarding path is also fixed. In this method, the route failure is detected if after predefined number of retransmissions the ACK for the frame is not received at the MAC layer. The motivation for node-level fixed gateway selection is that, it may help to balance data traffic load to multiple gateways, and it may also avoid longer data forwarding paths. Moreover, less frequent broadcasting of control messages due to the fixing of path and gateway can result in lower control overhead. Hereafter, we refer this method as node-based best gateway selection (NBS). Algorithm 3 summaries the gateway selection and packet forwarding for NBS.

3.3 Channel Capacity and Contention-based Routing Protocol

In this section, we present details of our routing protocol design. The routing protocol has the following three components:

Algorithm 3: Node-Level Fixed Gateway Selection and Packet Forwarding

```
selected_gateway = Null;
downstream_node = Null;
if data_pkt.src_addr == node_addr then
    selected_gateway = get_node_gateway();
    if selected_gateway == Null then
        selected_gateway = select_best_gateway();
    end
    data_pkt.gateway_addr = selected_gateway;
    downstream_node = get_downstream_node_from_G_Table(selected_gateway);
    relay_pkt(data_pkt, downstream_node);
end
else
    if data_pkt.gateway_addr ≠ node_addr then
        downstream_node = get_downstream_node_from_G_Table(data_pkt.gateway_addr);
        relay_pkt(data_pkt, downstream_node);
    end
end
```

- Route discovery and maintenance
- Loop-free routing
- Route repair

3.3.1 Route Discovery and Maintenance

The route discovery starts with a broadcast of the gateway INFO message by a gateway. The gateway INFO message is used to inform direct neighbours of the gateway about the gateway's presence. The INFO message broadcasted by the gateway contains the following information: message type, network layer address of the gateway, and gateway sequence number. The gateway periodically broadcasts the INFO message, and increments the sequence number. Direct neighbours of the gateway receives the INFO message, and update their G_{table}, if required. A gateway's direct neighbours store their own $Node_{cap}$ in the minimum capacity and minimum capacity with the impact of contention fields of G_{table}. If a node receives the INFO message from a gateway that was previously unknown, it adds a new record in its G_{table}, and stores the gateway's address, sequence number, hop-count and information related to the capacity. If the information about the gateway is already present in the node's G_{table}, and the sequence number received in the INFO message is greater than the sequence number stored in the relevant record of its G_{table}, the node updates the sequence number field in its G_{table}. Otherwise, the node ignores the INFO message. Whenever the node updates its $Node_{cap}$, it updates minimum capacity and minimum capacity with the impact of contention fields in its G_{table} for those gateways which are direct neighbours of the node. If the node does not receive the INFO message from the gateway before a time out interval, it removes the gateway record from its G_{table}. Algorithm 4 summarizes the processing of the gateway INFO message at direct neighbours of the gateway.

To disseminate the routing information to all the nodes in a network, periodically nodes broadcast the node HELLO message. The HELLO message contains information about the gateways discovered by a node, and the following is the specific information contained in the

Algorithm 4: Processing of the Gateway INFO Message

```
gateway_record = Null;
gateway_record = is_gateway_in_G_Table(info.gateway_address);
if gateway_record == Null then
    gateway_record = add_new_record_in_G_Table(info, Node_cap);
    gateway_record.set_time_out(timeout_value);
end
else
    if gateway_record.seq_no < info.seq_no then
        gateway_record.seq_no = info.seq_no;
        gateway_record.refresh_time_out();
    end
end
```

message: message type, number of gateways whose information is contained in the message, and the routing information to the gateways. The routing information corresponding to each gateway is: gateway address, gateway sequence number, minimum capacity on the path to the gateway, and hop-count. The minimum capacity on the path to a gateway i is represented by $Path_{cap_i}$. N_i represents a set of nodes on the path to gateway i, and a node in N_i is represented by j, therefore $Path_{cap_i} = min(Node_{cap_j} \forall j \in N_i)$.

When any node n receives the node HELLO message, for each gateway's information in the message the node repeat the following. If the gateway is not present in the node's G_{table}, the node inserts another record in its G_{table}. The minimum capacity on the path to the gateway field is set to $Path_{cap_i}$, the minimum capacity on the path with contention field is set to $Path_{cc} = \left(\frac{Path_{cap_i}}{AC}\right)$, the downstream node address is set to the HELLO message broadcasting node's address, hop count value present in the HELLO message is incremented by 1 and the hop count field in G_{table} is set equal to it, and the remaining fields of the G_{table} record are copied with the corresponding fields of the HELLO message. If the gateway is present in G_{table}, the node compares the gateway sequence number received in the message with the gateway sequence number stored in G_{table}. If the received sequence number is greater than the stored sequence number, the node replaces the minimum capacity on the path field with $Path_{cap_i}$, the minimum capacity on the path with contention with $Path_{cc}$, the downstream node address with the HELLO message broadcasting node address, hop count field with 1 plus the hop count value received in the message. The remaining fields in the relevant record of G_{table} are replaced with the corresponding fields of the HELLO message. Afterwards, the G_{table} record expiry timer is refreshed. If the received sequence number if less than the sequence number stored in G_{table}, the message is ignored. If a node does not receive the gateway information with higher sequence number before the time out, the relevant record from G_{table} is deleted. Algorithm 5 summarizes the processing of the node HELLO message.

3.3.2 Loop-Free Routing

Loops in routing can be harmful even if they are short-lived. Therefore, we use gateway sequence number to ensure loop-free routing. We illustrate the loop-free characteristic of our routing protocol with the help of an example. Consider a line topology of five nodes $A \leftrightarrow B \leftrightarrow C \leftrightarrow D \leftrightarrow E$. In the example node E is the gateway. Let us suppose that the communication link between node C and D breaks, therefore node C does not receive the node HELLO message

Algorithm 5: Processing of the Node HELLO Message

```
gateway_record = Null;
for i = 0 ; i < hello.no_of_gateways ; i = i + 1 do
    AC = min(((hello.gateway[i]).hop_count + 1), 5);
    gateway_record.capacity = min((hello.gateway[i]).path_cap, Node_cap);
    gateway_record.capacity_with_contention = gateway_record.capacity / AC;
    gateway_record.hop_count = (hello.gateway[i]).hop_count + 1;
    gateway_record.downstream_node = hello.broadcasting_node;
    if is_gateway_in_G_Table((hello.gateway[i]).address) == Null then
        gateway_record.seq_no = (hello.gateway[i]).seq_no;
        gateway_record.set_time_out(timeout_value);
        add_new_record_in_G_Table(gateway_record);
    end
    else
        gateway_record = get_record_from_G_Table((hello.gateway[i]).address);
        if (hello.gateway[i]).seq_no > gateway_record.seq_no then
            gateway_record.seq_no = (hello.gateway[i]).seq_no;
            update_record_in_G_Table((hello.gateway[i]).address, gateway_record);
            gateway_record.refreah_time_out();
        end
        else
            continue;
        end
    end
end
```

from node D. But, node B advertises the route in the HELLO message, as node B can only learn the route to node E through node C, therefore the sequence number known to node B can not be greater than the sequence number known to node C. Therefore, node C discards the HELLO message received from node B. Otherwise, node C could have created a loop by replacing the downstream node's address to node E in its G_{Table} with node B's address. If the link between node C and node D does not become available before the time out interval, the records corresponding to the gateway node E shall be deleted from G_{Table} of nodes A, B, and C.

3.3.3 Route Repair

A node infers a route failure in the following cases: (i) direct neighbours of a gateway does not receive the gateway HELLO message for a predefined interval of time, (ii) before a predefined interval of time the node HELLO message is not received from a node that is being used as the downstream node to a gateway, and (iii) ACK at the MAC layer is not received after a predefined number of retransmissions of the same data frame. In case of the route failure, a node uses a route failure message to inform upstream node about the failure, and in this way the information is communicated to a source node. Afterwards, the source node can select a different gateway, if available.

4 Performance Evaluation

In this section, we present our proposed protocols performance results in comparison with per-packet, per-flow, and node-level fixed random and closest gateway selection methods. We obtain the results through simulations and testbed-based experiments. For a comprehensive performance evaluation, we study the impact of the following on the different protocols performance: (a) number of gateways, (b) physically uniformly distributed transmitters, (c) a few non-uniformly distributed transmitters, (d) frequency of control messages, and (e) duty-cycling. As the performance of our proposed protocols is compared against random and closest gateway selection methods, therefore here we briefly describe the methods.

For random gateway selection, we use three different strategies: per-packet random gateway selection (PPRS), per-flow random gateway selection (FRS), and per-node random gateway selection (NRS). In PPRS, whenever a source node has a new data packet to transmit, it randomly selects a gateway for the packet from the list of discovered gateways. In FRS, whenever a source node initiates a new flow, it randomly selects a gateway for the flow and all the packets corresponding to the flow are forwarded to the same gateway. In NRS, when a source node is about to transmit its first data packet, it randomly selects a gateway and forwards all data packets originating from the node to the same gateway for the node's lifetime (unless there is a route or gateway failure). The random selection methods use the same routing protocol as used by our proposed protocols. In closest gateway selection (CGS), each source node uses the hop-count metric to select the closest gateway. The CGS method, uses shortest hop-count routing protocol.

In this section, we categorize our performance benchmarks as follows: reliability, latency, energy consumption, and fairness. For reliability we measure and report mean PDR, for latency we report mean per-packet end-to-end delay, for energy consumption we measure mean forwarding path length, total retransmissions, load imbalance in a network (calculated through Eq. 3), and protocols' control overhead. Finally, we measure fairness through mean fairness index as shown in Eq. 4.

$$L_{imb} = \sum_{i=1}^{N} (|PB - AL_i|) \tag{3}$$

We use Eq. 3 to measure the load imbalance in a network. L_{imb} represents total load imbalance, and a higher value of L_{imb} indicates poor load balancing. The poor load balancing may result in hotspots near different gateways, hence nodes near such gateways may deplete their energy faster. In Eq. 3, PB is the perfect balancing index; if there are two gateways in a network, ideally 50% of the total data traffic should be forwarded to each gateway, hence in this case PB is 50%. Similarly, PB values for 3 and 4 gateways are 33.33% and 25% respectively. AL_i represents the actual proportion of total data packets forwarded by different source nodes to gateway i. N is the total number of gateways.

$$F_{index} = 1 - \left(\frac{\left(\sum_{i=1}^{T} (PDR_h - PDR_i) \right)}{T} \right) \tag{4}$$

In Eq. 4, F_{index} represents mean fairness index, and a higher value of F_{index} indicates better fairness. PDR_h represents PDR of a node that achieved highest PDR in an experiment among all the nodes in a network. PDR_i is the PDR achieved by i^{th} node, and T is the total number of nodes (apart from gateways) in a network.

Table 1: General Simulation Parameters

Parameter	Value
MAC layer	Unslotted CSMA-CA
MAC layer reliability	Enabled
Radio duty cycling algorithm	ContikiMAC
Radio model	Unit disk graph model: distance loss
MAC layer queue size	10 frames
Channel rate	250 kbps
Node transmission range	50 meters
Node carrier sensing range	100 meters
Total frame size	127 bytes
Node Control Message Interval	1 second
Emulated mote	Tmote Sky

4.1 Simulation Results

We performed extensive simulation experiments, and our simulations were performed using the widely used Cooja wireless sensor network simulator that uses real programming code for a wireless sensor node [24]. In our simulations, we used a grid network topology with 100 sensor nodes, and the network spans an area of $450 \times 450m^2$. To analyse the impact of number of gateways on the gateway selection methods, we vary the number of gateways in the network from 2 to 4, and in each experiment gateways are randomly placed in the network. Our results are based on 10 simulations runs (randomly placing gateways each time) for each number of gateways. In the following figures, we plot the mean value for each gateway selection protocol, and we show as error bars the 95% confidence intervals (CIs) around the mean, based on t-distribution with a sample size of 10. We not only present results related to PDR, delay, total retransmissions, and control overhead, but we also present results related to load imbalance and path length. The results related to load imbalance and path length helps us to draw insightful observations about the methods. For PDR, delay, total retransmissions, and control overhead where CIs overlap and means are not in the overlap region, we base our conclusions on the results of a t-test. General simulation parameters are given in Table 1, and use based on those by other researchers.

4.1.1 Uniformly Distributed Transmitters

The purpose of experiments described in this sub-section is to determine the impact of the different gateway selection methods on reliability, latency, energy consumption, and fairness in the presence of physically uniformly distributed transmitters relative to the gateways. Moreover, we are also interested in investigating the methods performance as we increase the number of gateways in such a set up. In our experiments, apart from gateways, every node in the network generates data packets, and the packet generation rate is randomly distributed in the range [1, 2] packets per second. The size of each data frame is 127 bytes. Each node generates packets using an on/off schedule; a node generates data packets for a duration randomly distributed in the range [5, 10] seconds, afterwards the node waits for a random duration of time distributed in the range [5, 10] seconds before generating packets again. Single on schedule is called a flow. No node generates packets after 100 simulation seconds. The total duration of a simulation is 115 seconds. Our traffic generation model is a representation of a range of event-detection and reporting systems, for example, fire detection, target tracking, etc.

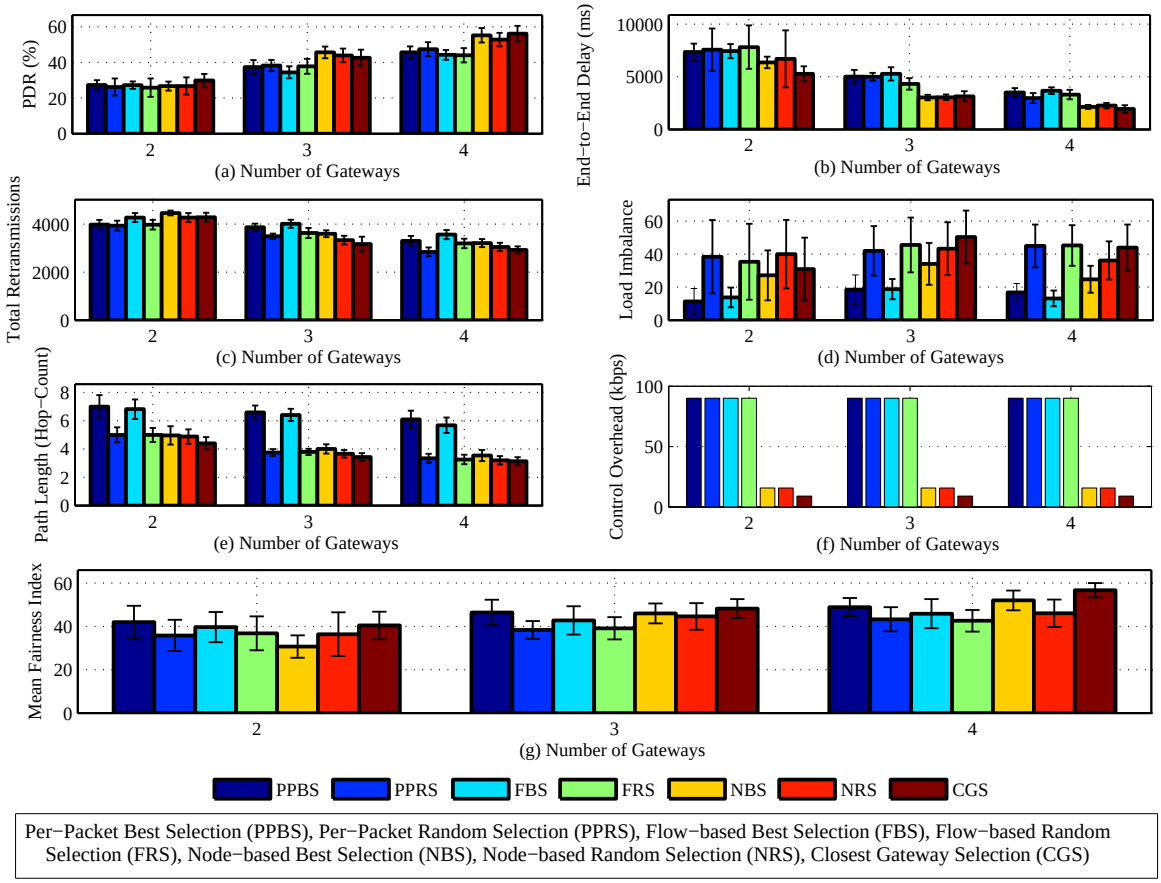


Figure 3: Protocols' Performance Comparison

Figure 3 shows the comparison of different protocols w.r.t. mean PDR, per-packet end-to-end delay, total retransmissions, load imbalance, forwarding path length, aggregate control overhead (control bits transmitted by all the nodes), and fairness. In general, with an increase in the number of gateways each protocol demonstrates higher PDR, lower delay, fewer retransmissions, and lower path length. The fewer retransmissions and lower path length imply lower energy consumption as we increase the number of gateways in the network. Despite statistically significantly lower load imbalance PPBS and FBS do not demonstrate higher PDR and lower delay as compared to the other protocols. This is due to the fact that both protocols select very lengthy paths, hence the advantage gained by lower load imbalance is wasted. Mostly, NBS, NRS, and CGS demonstrate higher PDR and lower delay as compared to the other protocols. Figure 3(e) demonstrates that in most cases PPRS, FRS, NBS, NRS, and CGS select paths with similar lengths, but only NBS, NRS, and CGS demonstrate better performance. In this case, the lower overhead of NBS, NRS, and CGS as shown in Figure 3(f) positively impacts the protocols' performance. Mostly among NBS, NRS, and CGS, on average CGS selects shorter paths, but it demonstrates similar performance in terms of PDR, delay, and total retransmissions. CGS is not superior as its mean load imbalance is higher as compared to NBS and NRS, the high load imbalance can create hotspots near the gateways. Hence, the hotspots near the gateways

can quickly deplete energy of nodes near the gateways. On the contrary, in most scenarios NBS demonstrates lower load imbalance and slightly higher path length, and still it demonstrates similar performance as compared to CGS. Therefore, NBS holds the advantage of significantly lower load imbalance over CGS by slightly increasing the path length, hence it better handles the hotspot problem. This reduces energy consumption of nodes near the gateways. Overall, only lowering load imbalance or path length does not result in an overall better performance of a gateway selection protocol, as demonstrated by the performance of PPBS and CGS respectively. NBS trades-off load imbalance and path length in a better way along with lower control overhead, therefore it not only demonstrates similar performance to CGS, but also reduces the hotspot problem. NBS does better load imbalance and path length trade-off because every node selects the best gateway when it has a first data packet to transmit. Generally, different nodes start their transmission at different times, hence better load balancing. Moreover, it avoids longer paths by not changing the path to a gateway.

Figure 3(g) shows mean fairness index of the protocols w.r.t. the number of gateways. Mostly, on average the random gateway selection protocols, i.e., PPRS, FRS, and NRS demonstrate lower fairness as compared to the other protocols. The reason being, the random protocols demonstrate higher load imbalance compared to the best gateway selection protocols, i.e., PPBS, FBS, and NBS as shown in Figure 3(d). The higher load imbalance creates higher contention near certain gateways, hence lower fairness. As per Figure 3(d) CGS has also demonstrated higher load imbalance, but its fairness is still higher than the random protocols. The reason being, mostly, CGS selects shorter paths as shown in Figure 3(e), hence CGS better trades-off load imbalance and path length. Although NBS selects slightly longer paths, but its fairness is similar to CGS. This is due to the fact that, NBS has demonstrated lower load imbalance. Therefore, the protocols that trade-off load imbalance and path length in a better manner demonstrate better fairness. In most cases, despite of selecting longer paths, PPBS demonstrates similar fairness as of NBS and CGS. The reason being, PPBS has demonstrated a lower load imbalance, and secondly longer paths create higher contention in the network, hence most of the nodes' PDR is affected (as mostly demonstrated by the lower PDR of PPBS), therefore PPBS appear to be a fair protocol.

4.1.2 Higher Control Message Frequency

To understand the impact of a higher frequency control messages on the protocols' performance, we carried out another set of simulations. In these experiments, each protocol doubles its control message broadcasting frequency, i.e., a control message is scheduled to be broadcasted every 500 ms. The details of our data traffic generation model are the same as described in Section 4.1.1. The results presented in Section 4.1.1 demonstrate that, the protocols demonstrate the worse and best performance in case of two and four gateways respectively. In this section, the aim is to analyse the impact of higher frequency control messages on the worse and best scenarios, hence we present the results pertaining to the two and four gateways scenarios.

Figure 4 shows the performance comparison of the protocols with the increased frequency of control messages. In this case, generally, the mean PDR of PPBS, PPRS, FBS, and FRS has lowered, and their delay has substantially increased as compared to their PDR and delay shown in Figure 3(a) and Figure 3(b) respectively. Moreover, in case of four gateways the protocols demonstrate higher number of retransmissions in comparison to the results shown in Figure 3(c). Apart from PPRS, these protocols demonstrate similar forwarding path lengths as compared to their path lengths show in Figure 3(e). Mostly, these protocols demonstrate lower load imbalance in comparison to their load imbalance shown in Figure 3(d). The lower load imbalance is due to the higher frequency of control messages, hence nodes get faster channel capacity information to the different gateways, and it helps in load balancing. In the light of above discussion, these

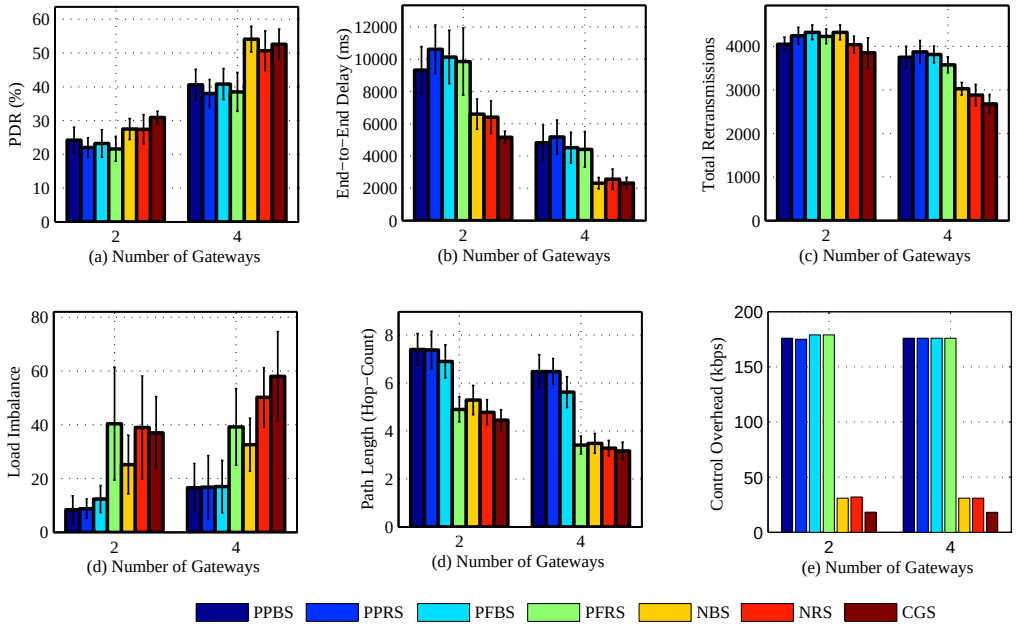


Figure 4: Protocols' Performance Comparison (Higher Control Message Frequency)

protocols' lower PDR and higher delay is due to the higher control overhead, as shown in Figure 4(f). Mostly, not only for these set of experiments, but in comparison to the results shown in Figure 3 NBS, NRS, and CGS demonstrate better performance. These protocols only transmit control messages for a specific duration of time, hence their lower overhead positively impact the performance. The control overhead corresponding to each protocol as shown in Figure 4(f) is not exactly twice the control overhead for the protocols shown in Figure 3(f), this is due to the fact that control messages are scheduled for broadcast every 500 ms, but the Contiki operating system's scheduler does not always schedule the task at the exact time instance. In general, the performance trend of NBS, NRS, and CGS are similar to Figure 3. Moreover, these results demonstrate that the higher frequency control messages slightly improves the performance of those protocols that only broadcast control messages for a limited time interval, whereas for the other protocol it negatively impacts the protocols' performance.

4.1.3 Non-Uniformly Distributed Transmitters

To investigate the protocols' performance when there are a few transmitters and most of them are close to a particular gateway, we performed another set of simulations. For these simulations, only six nodes act as the source nodes. Most of the source nodes are placed closer to one gateway. The data generation rate of the nodes is randomly distributed in the range [5, 7] packets/second. The nodes generate data packets using an on/off schedule; the nodes generate packets for a duration randomly distributed in the range [5, 20] seconds, afterwards the nodes wait for a time randomly distributed in the range [5, 10] seconds before generating packets again. As there are only six nodes generating data, therefore for a reasonable data activity in the network, we increased the nodes' data generation rate and the amount of time for which the nodes generate data as compared to the traffic model described in Section 4.1.1. Protocols' control messages

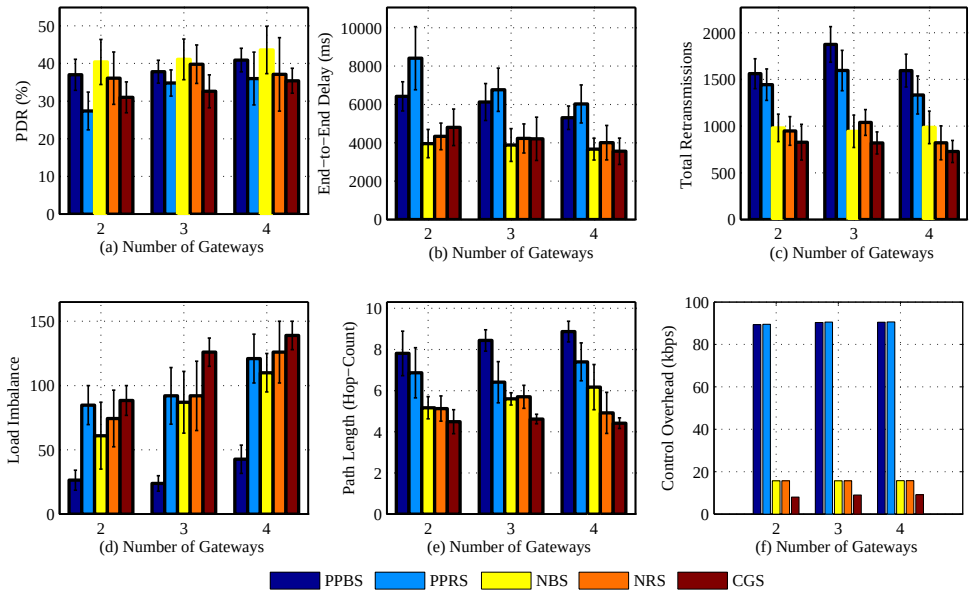


Figure 5: Protocols' Performance Comparison (Non-Uniformly Distributed Transmitters)

are broadcast every second. Generally, in Section 4.1.1 among channel capacity and contention-based gateway selection protocols, per-packet and node-based selection protocols demonstrated worst and best performance respectively, therefore in this section we only analyse the impact of non-uniform transmitters on the worst and best performing protocols.

Figure 5 shows the protocols' performance comparison. The results demonstrate the following different trends in comparison to the results shown in Figure 3: In all scenario, channel capacity and contention-based best gateway selection protocols demonstrate higher PDR as compared to CGS, in all scenarios NBS and CGS demonstrate similar delay though mostly NBS demonstrates lower mean, and mostly NBS has selected longer paths. Due to the excessively higher load imbalance as shown in Figure 5(d), CGS demonstrates inferior performance. Apart from the above, mostly other performance trends are similar to the trends shown in Figure 3.

4.1.4 No Duty Cycling

To obtain an upper bound on the protocols' performance, we performed another set of simulations without using radio duty cycling. Therefore, for these simulations we used Contiki's Null radio duty cycling algorithm. Data traffic generation model and all other simulation parameters are the same as presented in Section 4.1.1.

Figure 6 presents the protocols' performance results under no duty cycling. The results are different in the following aspects as compared to the results presented in Figure 3:

- Each protocol's PDR has substantially improved.

  . No duty cycling results in a higher number of chances for data transmission, hence fewer packet drops at the MAC layer queue. These factors contribute to the protocols' improved PDR.

- Delay for each protocol has immensely decreased.

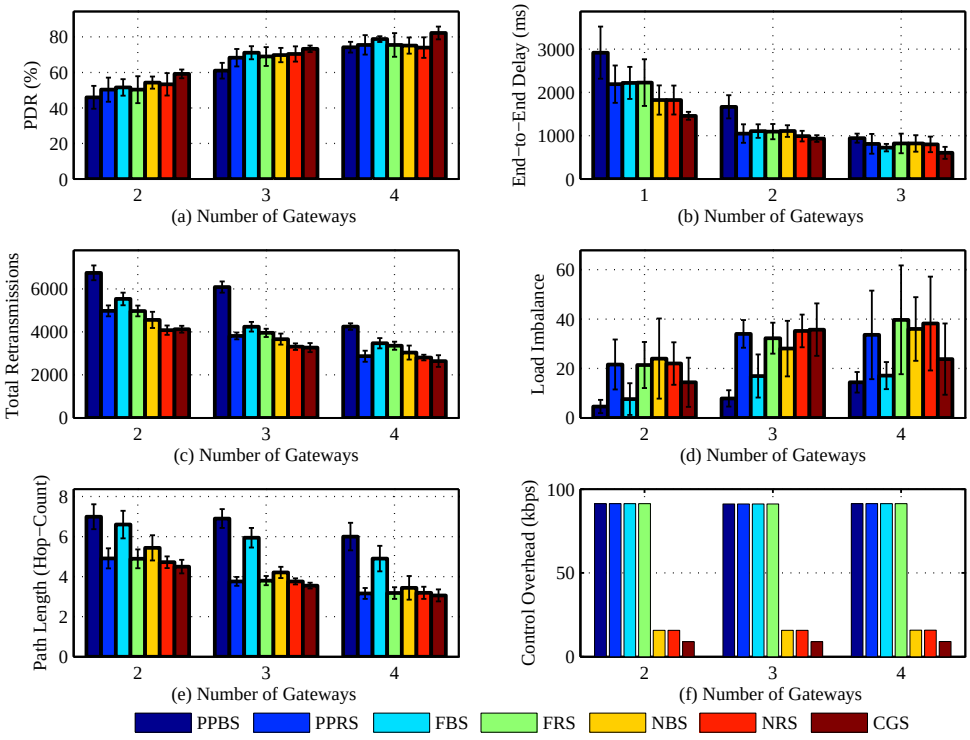


Figure 6: Protocols' Performance Comparison (No Duty Cycling)

. More frequent successful data packet transmission results in lower delay.

- Mostly, apart from PPBS all protocols demonstrate similar PDR and delay.

  . The results obtained under duty cycling demonstrated the impact of control overhead on the protocols' performance, but in this case overhead does not seem to have a great impact on the protocols' performance (given that in both cases, i.e., with and without duty cycling path lengths are similar). Therefore, similar performance of the protocols.

- Mostly, each protocol demonstrates a higher number of retransmissions.

  . The higher number of retransmissions with higher PDR implies that the higher retransmissions are due to a higher data activity. Duty cycling can result in filling up the MAC layer queue, hence packets are dropped without any transmission attempt, therefore lower number of retransmissions under duty cycling operation.

- CGS demonstrated lower load imbalance as compared to the load imbalance demonstrated by the protocol under duty cycling operation.

  . As PDR of each protocol has substantially increased under no duty cycling operation, this implies that the percentage of control packets received by the nodes are high as well. This helps each node to discover all gateways in a timely manner, and this is

important for load balancing in CGS as after sometime the protocol does not transmit broadcast messages.

Mostly, under no duty cycling operation NBS, NRS, and CGS demonstrate similar performance, but CGS demonstrate better mean for different performance metrics. Moreover, lower load imbalance, path length, and fewer total retransmission imply lower energy consumption, therefore overall, CGS demonstrates better performance followed by NBS. Comparison of the results obtained using the duty cycling algorithm and without duty cycling reveal that, due to the lower transmission opportunity in duty-cycling-based operation the higher control overhead impacts the protocols' performance, whereas it does not seem to impact the protocols' performance when there is no duty cycling.

Generally, devices operate under duty cycling, and our results demonstrate that under duty cycling regardless of the uniformity and non-uniformity of the transmitters NBS makes a better trade-off between load imbalance and path length, hence it demonstrates better performance as compared to the other protocols. PPBS substantially improves load imbalance, but selecting far away gateways negatively impacts the protocol's performance when the source nodes are uniformly distributed. But, still it performs better than CGS when the source nodes are non-uniformly distributed. Furthermore, only minimizing path length or load imbalance does not result in a good joint routing and gateway selection algorithm.

4.1.5 Comparison with Available-Bandwidth-based Protocol

In previous sections, we compared our protocols' performance against random and closest gateway selection methods. As discussed in Section 2, there are joint routing and gateway selection protocols that try to enhance a network's throughput. The available bandwidth on the path to a particular gateway gives an indication of the achievable throughput to the gateway. A number of available-bandwidth-based routing protocols for delay-sensitive IEEE 802.15.4-based ad-hoc networks has been designed and evaluated in [25]. It has been demonstrated that, the protocol that uses the available bandwidth and shortest hop-count as the composite routing metric performs better than the other evaluated protocols. The routing protocol selects candidate forwarding paths to a sink based on the shortest hop-count, and if there are multiple such paths, the protocol selects the path that has the highest end-to-end available bandwidth. Therefore, in this section we compare our protocols' performance against the routing protocol that used the available bandwidth and shortest hop-count as the composite metric.

Simulations were performed using Cooja simulator. We used a grid network topology with 75 nodes placed in a $300 \times 300\, m^2$ area. Each node generates data packets, and the packet generation rate is randomly distributed in the range [1, 3] packets/second. The size of data frame is 127 bytes. Nodes generate data packets using an on/off schedule, i.e., the nodes generate the packets for a duration randomly distributed in the range [2, 5] seconds, afterwards the nodes wait for a random duration of time distributed in the range [10, 15] seconds before generating packets again. No node generates packets after 100 simulation seconds. The total duration of a single simulation is 115 seconds. Our results are based on 10 simulation runs (randomly placing gateways each time) for each number of gateway nodes. For these experiments, we do not use any radio duty cycling algorithm. Simulation parameters are given in Table 1.

Figure 7 shows comparison of the protocols in terms of mean PDR, mean end-to-end delay, and total number of retransmissions. The figure shows that, our protocols demonstrate higher mean PDR as compared to the available-bandwidth-based protocol. Our protocols demonstrate up to 37% lower mean end-to-end delay as compared to the available-bandwidth-based protocol. Moreover, our protocols demonstrate up to 48% lower total number of retransmissions as compared to the available-bandwidth-based protocol. It is demonstrated in [25] that, apart from

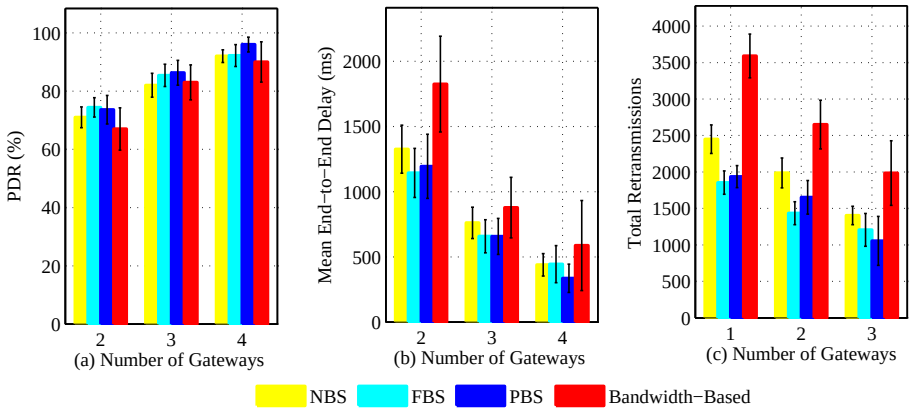


Figure 7: Protocols' Performance Comparison with Available-Bandwidth-Based Protocol

the regular control messages required for routing, additional control messages are also required to estimate the available bandwidth. Our joint routing and gateway selection protocols do not require extra control messages apart from the regular messages required for routing. Hence, our protocols' control overhead is also lower than the available-bandwidth-based protocol. Therefore, our protocols are not only better in terms of PDR and delay, they are also energy-efficient than the available-bandwidth-based protocol due to lower total retransmissions and control messages overhead.

4.1.6 Results with Modified Capacity and Contention Metric

The maximum contention count on a node along a forwarding path is 5, i.e., two contention counts due to upstream nodes, two contention counts due to downstream nodes, and as a node itself also relays/generate data, therefore a single contention count for the node itself is required. Hence, the maximum contention count is 5, and hereafter we refer to it as MAX_COUNT. If the path length to a gateway is greater than 5, there are multiple nodes along the path whose contention count is MAX_COUNT. This number is equal to $(hop_count - MAX_COUNT)$. Therefore, in our modified metric, we consider its impact on the capacity. To accomplish this, we use the relation $\left(\frac{Node_{cap}}{hop_count}\right)$ bps, if hop-count to a gateway is less than or equal to 5. Otherwise, $\left(\frac{Node_{cap}}{MAX_COUNT+(hop_count-MAX_COUNT)}\right)$ bps. This relation also simplifies to $\left(\frac{Node_{cap}}{hop_count}\right)$ bps.

We modified our joint routing and gateway selection protocols using the modified metric. The modified protocols are called node-based best selection modified (NBS-Modified), flow-based best selection modified (FBS-Modified), and packet-based best selection modified (PBS-Modified). For performance evaluation, we performed another set of simulation-based experiments. The data generation model and other simulation details are the same as given in Section 4.1.5.

Figure 8 shows comparison of the protocols in terms of mean PDR, mean end-to-end delay, and total retransmissions. In general, the modified protocols demonstrate similar performance. Comparison of Figure 8 and Figure 7 shows that in general, the modified version of our protocols show similar mean PDR and lower delay and total retransmissions. The better performance is due to the fact that, if different paths advertise similar capacity and the paths' lengths are greater than 5, our original protocols consider them equally good. But, our modified protocols prefer shorter path among the available paths. As our modified protocols slightly improve the performance of our original protocols and our original protocols demonstrate better performance

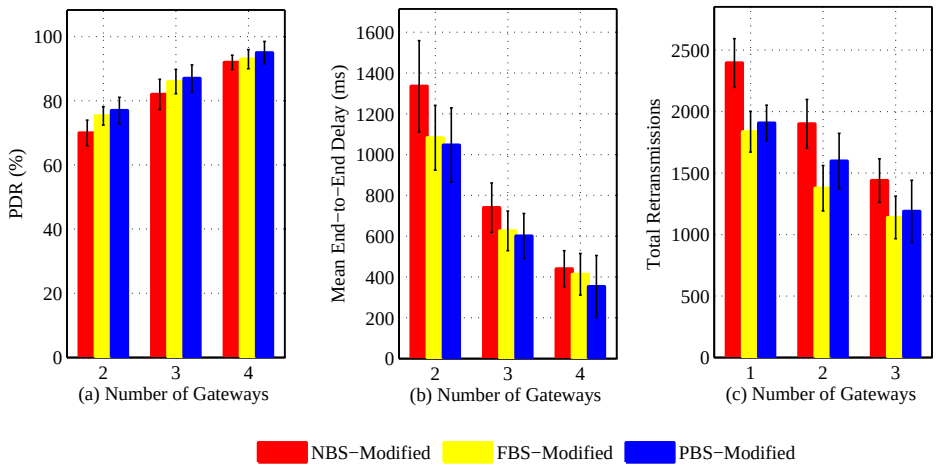


Figure 8: Protocols' Performance Comparison with Modified Metric

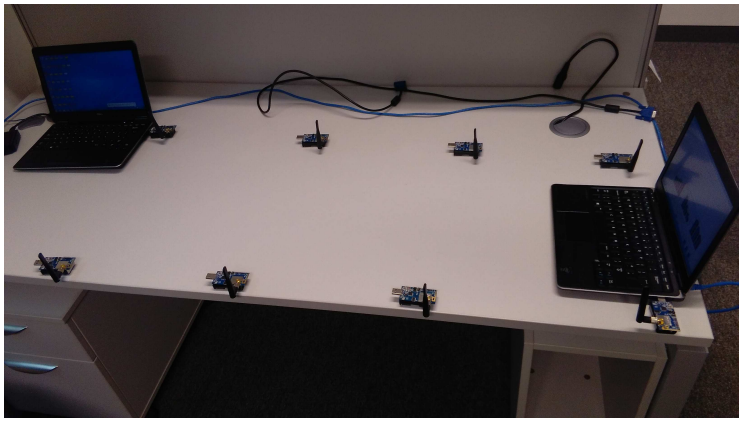


Figure 9: Testbed Topology

than the available-bandwidth-based protocol. Therefore, our modified protocols are also better than the available-bandwidth-based protocol.

4.2 Testbed Results

To validate simulation results, we deploy a network consisting of 8 TelosB motes as shown in Figure 9. The distance between direct neighbour nodes is approximately 0.5 m. The transmission power of the nodes' radios are carefully adjusted so that these nodes approximately form a network with the grid topology. The nodes attached to the laptops shown in Figure 9 act as the gateways. The traffic generation model and the radio duty cycling algorithm are the same as presented in Section 4.1.1 and Section 4.1.4 respectively.

Figure 10 shows the comparison of the protocols' performance with two gateways. The PDR results shown in Figure 10(a) are mostly consistent with the PDR results for the 2 gateways scenario shown in Figure 6(a). Mostly, all protocols demonstrate similar PDR, but NBS and

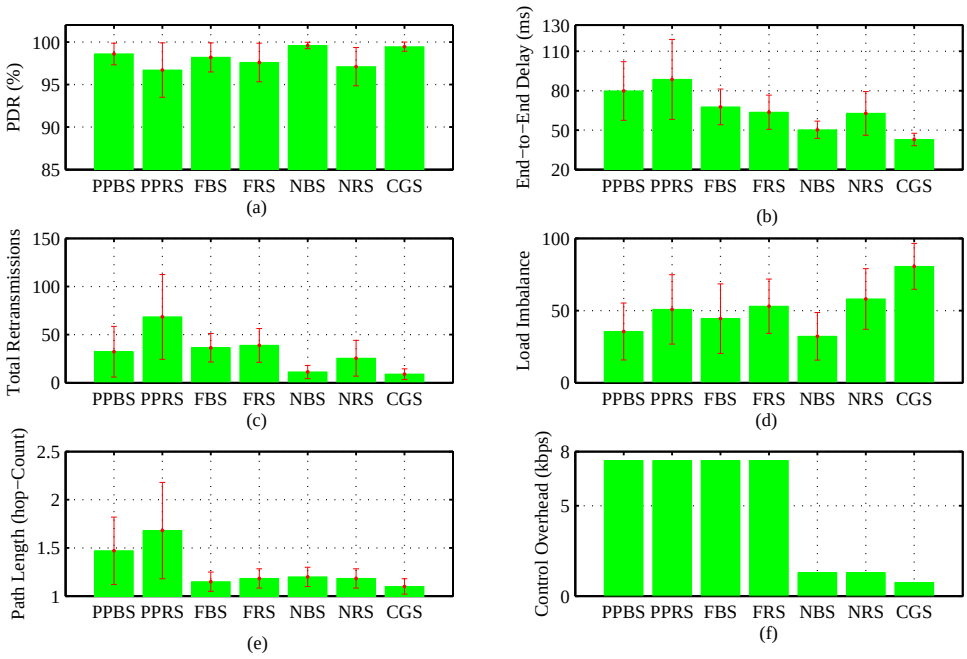


Figure 10: Testbed-based Protocols' Performance Comparison

CGS demonstrate higher mean value. All protocols demonstrate higher mean PDR as compared to the results presented in Figure 6(a), and this is because of the fact that the testbed consists of 8 nodes whereas in simulation there were 100 nodes. In the testbed, lower traffic in the network improves the protocols' overall PDR. Mostly, the delay and total retransmissions results demonstrate similar trends as shown in Figure 6(b) and 6(c) respectively, i.e., NBS, NRS, and CGS demonstrate lower mean as compared to the other protocols. The load imbalance in this case demonstrate different trend as compared to the results presented in Figure 6(d). NBS demonstrates lower load imbalance, and it is even lower than PPBS; the protocol that demonstrated lowest load imbalance in Figure 6(d). Again, this is due to the fact that the testbed consists of a smaller number of nodes as compared to the simulation setup. In this case. mostly, path length and control overhead demonstrate similar trends as shown in Figure 6(e) and Figure 6(f) respectively. In general, our testbed results validate the simulation results.

4.3 Discussion

While choosing a joint routing and gateway selection protocol for machine-type communication under duty-cycling operation, one should choose a protocol that has a lower control overhead and that can make a good trade-off between the forwarding path length and load-imbalance. Our results demonstrate that NBS possesses the mentioned characteristics. Under no duty-cycling operation, control overhead does not impact the performance, but it results in more energy consumption. Increasing the frequency of control messages negatively impacts those protocols' performance who continue to periodically transmit control messages, but the higher frequency of control messages positively impact those protocols' performance who broadcast control messages

for a short time interval, for example, NBS and CGS. In case most of the transmitters are close to a particular gateway, CGS should be avoided. Invariably, adding more gateways to a network results in a better performance, therefore if further improvement in the performance is required, one should consider increasing the number of gateways in a network.

5 Conclusions

Using the end-to-end channel capacity and contention metrics, we designed joint routing and gateway selection methods for MTC. Based on the metrics, our methods selected the best gateway on per-packet, per-flow, and per-node decision granularity. Our evaluation demonstrated that selecting the best gateway on per-packet basis is best in load-balancing to different gateways, but it selects longer paths. Node-based best gateway selection selects shorter paths, but the load-balancing is poorer as compared to the per-packet selection. Moreover, node-based gateway selection demonstrated lower control overhead. Flow-based best gateway selection is good in load-balancing, but it selects longer paths as compared to node-based selection. For performance evaluation, we also compared our methods with existing gateway selection methods. We specifically, analysed the impact of the number of gateways, physical distribution of transmitters, control overhead, and duty-cycling on the performance of the selection methods. Our results demonstrated that, with an increase in the number of gateways the methods' performance improve, control overhead only impacts the methods' performance in duty-cycled operation, closest gateway selection did not demonstrate good results when the transmitters are close to a certain gateway, and the performance of the method depends on making a good trade-off between load imbalance to different gateways and a forwarding path's length. Mostly, our results demonstrated that, NBS makes a better trade-off between load imbalance and path length, hence generally it outperformed other methods. Using a static network topology our PPBS, FBS, and NBS methods demonstrated up to 325%, 67%, and 33% better load balancing respectively as compared to CGS. PPBS and NBS demonstrated up to 22% and 13% higher PDR respectively as compared to the CGS method. Moreover, our methods demonstrated up to 30% lower mean end-to-end delay and up to 48% lower total number of retransmissions as compared to the state-of-the-art available-bandwidth-based method.

References

[1] 3GPP, *Service Requirements for Machine-Type Communications. Technical report, TR 22.368, 2012.*

[2] S. Pellicer, G. Santa, A. L. Bleda, R. Maestre, A. J. Jara, and A. G. Skarmeta, "A Global Perspective of Smart Cities: A Survey," in *International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2013, pp. 439–444.

[3] "LTE-M Optimizing LTE for the Internet of Things," White Paper, Nokia, 2015.

[4] M. Condoluci, M. Dohle, G. Araniti, M. A, and K. Zheng, "Toward 5G Densenets: Architectural Advances for Effective Machine-Type Communications Over Femtocells," *IEEE Communications Magazine*, vol. 53, no. 1, pp. 134–141, 2015.

[5] S. T. Cheng and T. Y. Chang, "An adaptive learning scheme for load balancing with zone partition in multi-sink wireless sensor network ," *Expert Systems with Applications*, vol. 39, no. 10, pp. 9427 – 9434, 2012.

[6] H. Yoo, M. Shim, D. Kim, and K. H. Kim, "GLOBAL: A Gradient-Based Routing Protocol for Load-Balancing in Large-Scale Wireless Sensor Networks with Multiple Sinks," in *IEEE Symposium on Computers and Communications*, 2010, pp. 556–562.

[7] H. Yoo, M. Shim, D. Kim, and K. H. Kim, "A Scalable Multi-Sink Gradient-Based Routing Protocol For Traffic Load-Balancing ," *Eurasip Journal of Wireless Communication and Networking*, vol. 2011, no. 85, pp. 1–16, 2011.

[8] H. Jiang and R. Sun, "Energy Optimized Routing Algorithm in Multi-Sink WSNs," *Applied Mathematics & Information Sciences*, vol. 8, no. 1, pp. 349–354, 2014.

[9] W. Liu, H. Nishiyama, N. Kato, Y. Shimizu, and T. Kumagai, "A Novel Gateway Selection Method to Maximize the System Throughput of Wireless Mesh Network Deployed in Disaster Areas," in 23^{rd} *IEEE International Symposium on Personal Indoor and Mobile Radio Communications*, 2012, pp. 771–776.

[10] W. Liu, H. Nishiyama, N. Kato, Y. Shimizu, and T. Kumagaii, "A Novel Gateway Selection Technique for Throughput Optimization in Configurable Wireless Mesh Networks," *International Journal of Wireless Information Networks*, vol. 20, no. 3, pp. 195–203, 2013.

[11] M. Ha, K. Kwon, D. Kim, and P. Y. Kong, "Dynamic and Distributed Load Balancing Scheme in Multi-gateway Based 6LoWPAN," in *IEEE International Conference on Green Computing and Communications, and Cyber Physical and Social Computing*, 2014, pp. 87–94.

[12] J. Feng, L. Zheng, J. Fu, and Z. Liu, "An Optimum Gateway Discovery and Selection Mechanism in WSN and Mobile Cellular Network Integration," in 8^{th} *International ICST Conference on Communications and Networking in China*, 2013, pp. 483–487.

[13] P. A. K. Acharya, D. L. Johnson, and E. M. Belding, "Gateway-Aware Routing for Wireless Mesh Networks," in 7^{th} *IEEE International Conference on Mobile Adhoc and Sensor Systems*, 2010, pp. 564–569.

[14] D. Kominami, M. Sugano, M. Murata, and T. Hatauchi, "Controlled Potential-based Routing for Large-scale Wireless Sensor Networks," in 14^{th} *ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2011, pp. 187–196.

[15] U. Ashraf, S. Abdellatif, and G. Juanole, "Gateway Selection in Backbone Wireless Mesh Networks," in *IEEE Wireless Communications and Networking Conference*, 2009, pp. 1–6.

[16] D. Nandiraju, L. Santhanam, N. Nandiraju, and D. P. Agrawal, "Achieving Load Balancing in Wireless Mesh Networks Through Multiple Gateways," in *IEEE International Conference on Mobile Adhoc and Sensor Systems*, 2006, pp. 807–812.

[17] V. G. T. N. Vidanagama, D. Arai, and T. Ogishi, "Service Environment for Smart Wireless Devices: An M2M Gateway Selection Scheme," *IEEE Access*, vol. 3, pp. 666–677, 2015.

[18] A. Cabrera, A. J. Yuste-Delgado, and D. Macas, "A Fuzzy Logic-Based and Distributed Gateway Selection for Wireless Sensor Networks," in *Highlights in Practical Applications of Agents and Multiagent Systems*, ser. Advances in Intelligent and Soft Computing, 2011, vol. 89, pp. 243–248.

[19] M. O. Farooq and T. Kunz, "Available-Bandwidth-Based Routing in IEEE 802.15.4-Based Ad-Hoc Networks: Proactive vs. Opportunistic Technique," in 28^{th} *IEEE International Conference on Advanced Information Networking and Applications*, 2014, pp. 57–64.

[20] E. Ancillotti, R. Bruno, and M. Conti, "Load-Balanced Routing and Gateway Selection in Wireless Mesh Networks: Design, Implementation and Experimentation," in *IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks*, 2010, pp. 1–7.

[21] M. O. Farooq and T. Kunz, "Impact of Route Length on the Performance of Routing and Flow Admission Control Algorithms in Wireless Sensor Networks," *IET Wireless Sensor Systems*, vol. 6, no. 1, pp. 10–16, 2016.

[22] S. Isik, M. Y. Donmez, and C. Ersoy, "Multi-Sink Load Balanced Forwarding with a Multi-Criteria Fuzzy Sink Selection for Video Sensor Networks ," *Computer Networks*, vol. 56, no. 2, pp. 615 – 627, 2012.

[23] M. O. Farooq and T. Kunz, "BandEst: Measurement-Based Available Bandwidth Estimation and Flow Admission Control Algorithm for Ad Hoc IEEE 802.15.4-Based Wireless Multimedia Networks," *International Journal of Distributed Sensor Networks*, vol. 2015, pp. 1–15, 2015.

[24] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross - Level Sensor Network Simulation with Cooja," in *31^{st} IEEE conference on Local Computer Networks*, 2006, pp. 641–648.

[25] M. O. Farooq, T. Kunz, C. J. Sreenan, and K. Brown, "Evaluation of Available Bandwidth as a routing Metric for Delay-Sensitive {IEEE} 802.15.4-based Ad-hoc Networks," *Ad Hoc Networks*, vol. 37, Part 2, pp. 526 – 542, 2016.