symmetry

MDPI

Article

QoS Enabled Layered Based Clustering for Reactive Flooding in the Internet of Things

Fawad Ali Khan [1,*], Rafidah Md Noor [1,2,*], Miss Laiha Mat Kiah [1], Noorzaily Mohd Noor [1], Saleh M. Altowaijri [3] and Atiq Ur Rahman [3]

[1] Department of Computer System & Technology, Faculty of Computer Science & Information Technology, University Malaya, Kuala Lumpur 50603, Malaysia; misslaiha@um.edu.my (M.L.M.K.); zaily@um.edu.my (N.M.N.)

[2] Centre of Mobile Cloud Computing Research (C4MCCR), Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur 59100, Malaysia

[3] Faculty of Computing and Information Technology, Northern Border University, Rafha 76321, Saudi Arabia; saltowaijri@nbu.edu.sa (S.M.A.); atiq621@gmail.com (A.U.R.)

* Correspondence: fawadkn@siswa.um.edu.my (F.A.K.); fidah@um.edu.my (R.M.N.)

Received: 15 April 2019; Accepted: 29 April 2019; Published: 5 May 2019

Abstract: The Internet of Things has gained substantial attention over the last few years, because of connecting daily things in a wide range of application and domains. A large number of sensors require bandwidth and network resources to give-and-take queries among a heterogeneous IoT network. Network flooding is a key questioning strategy for successful exchange of queries. However, the risk of the original flooding is prone to unwanted and redundant network queries which may lead to heavy network traffic. Redundant, unwanted, and flooded queries are major causes of inefficient utilization of resources. IoT devices consume more energy and high computational time. More queries leads to consumption of more bandwidth, cost, and miserable QoS. Current existing approaches focused primarily on how to speed up the basic routing for IoT devices. However, solutions for flooding are not being addressed. In this paper, we propose a cluster-based flooding (CBF) as an interoperable solution for network and sensor layer devices which is also capable minimizing the energy consumption, cost, network flooding, identifying, and eliminating of redundant flooding queries using query control mechanisms. The proposed CBF divides the network into different clusters, local queries for information are proactively maintained by the intralayer cluster (IALC), while the interlayer cluster (IELC) is responsible for reactively obtain the routing queries to the destinations outside the cluster. CBF is a hybrid approach, having the potential to be more efficient against traditional schemes in term of query traffic generation. However, in the absence of appropriate redundant query detection and termination techniques, the CBF may generate more control traffic compared to the standard flooding techniques. In this research work, we used Cooja simulator to evaluate the performance of the proposed CBF. According to the simulation results the proposed technique has superiority in term of traffic delay, QoS/throughput, and energy consumption, under various performance metrics compared with traditional flooding and state of the art.

Keywords: QoS; redundant query; Internet of things; network flooding; energy efficiency

1. Introduction

The Internet of Things (IoT) has become quite famous in the recent years. Many of our daily routine devices are getting connected with us, covering many capabilities like sensing, autonomy, and contextual awareness [1]. The IoT results from Internet progress and the innovative evolution of

smart devices has led to the development of the new computing prototype. IoT is the next revolutionary technology in converting the present communication infrastructure into a completely future network [2]. IoT is expected to contain very large numbers of sensors collecting and passing on data on environmental conditions, physiological measurements, machine operational data, etc. IoT provides an integration of various sensors and objects that can communicate directly with one another without human intervention [3,4].

The primary purpose of the IoT is to allow a secure data exchange between the real world devices and applications [5]. IoT is, therefore, based upon the integration of several communication solutions, identification and tracking technologies, sensor and actuator networks, and distributed smart objects [6]. These objects/devices are connected to each other and share the same network for communicating with each other. These devices are connected with the sensor to detect the particular surrounding conditions and analyze the situation and work accordingly. IoT devices are also programmed to make decisions automatically or inform [7] , according to the user, so that the user can make the best decision. This interconnected network can bring a great deal of advancement in the technology of applications and services which can bring economic benefit to global business development. Many devices are connected to the Internet to share the local information in cyberspace.

Moreover, IoT promises a smart environment that would offer immense savings of time, energy, good quality of service (QoS), and less delay and resources [8]. Dynamic resource scheduling for heterogeneous workloads in IoT is critical for ensuring QoS, level of energy consumptions on each mote, and the traffic delay during data transmission [9]. Energy consumptions, QoS, and delay are the major challenging requirements for IoT networks since data transmission in the IoT network is based on priority [10]. Additionally, it is a known fact that IoT has the potential for a wide range of applications relating to agriculture, transportation, health, education, supply chain, farming, plant disease diagnosis, poultry, irrigation, and pest control [11] and each application requires a large number of sensors to connect and communicate with each other, which may reduce the QoS of the network due to inefficient resource utilization, traffic delay due to redundant messages/queries because each device has direct access to cloud resources, and energy consumption [12,13], as illustrated in Figure 1.

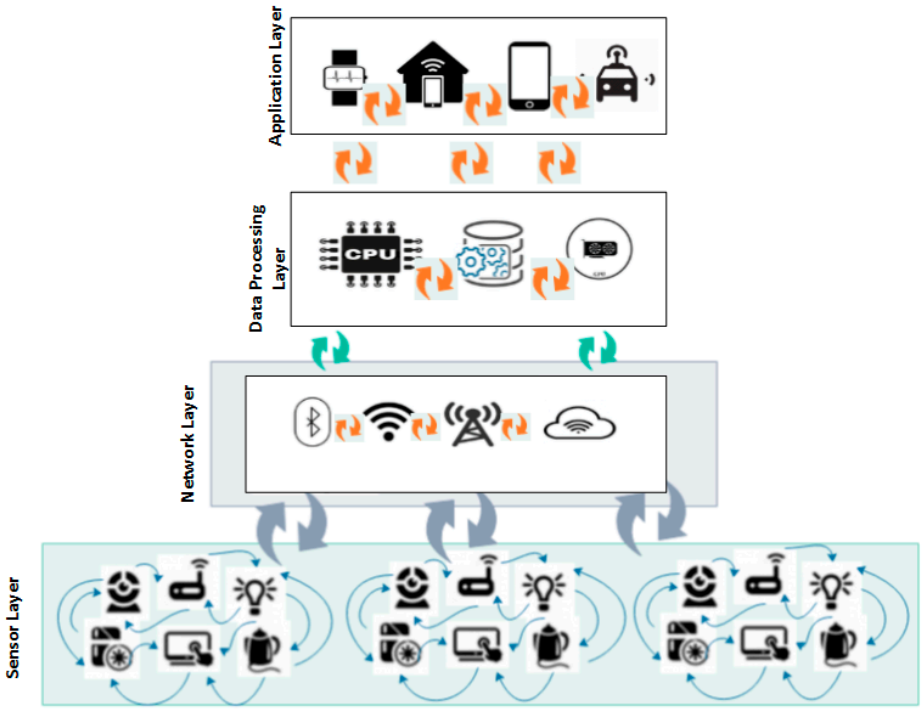

Figure 1. Layer-based system model with different motes communicating redundantly in IoT.

Figure 1 shows the system model with different sensor motes communicating with each other between the physical (sensor) and network layers of IoT. From the figure it can be seen that redundant messages, unwanted queries and network flooding are the major causes of inefficient utilization of resources, thus making IoT devices consume more energy and high computational time (i.e., delay in data transmission), which, in turn, affects the network QoS [14]. Moreover, solving these issues in IoT networks is demanding due to the constraint nature of the devices with limited energy. Presently, to the best of our knowledge, no mechanisms for identification of redundant queries have been developed in this domain.

Following are the contributions of this paper: We examine several approaches for tackling unwanted and redundant communication in the IoT network to enable us to understand the sequence of actions that take place when the flooding are happening and propose the cluster-based flooding (CBF) technique. The proposed technique is an interoperable solution both for physical layer and network layer devices. CBF divides the network into different clusters, local queries are proactively maintained by the intralayer clustering (IALC), while interlayer clustering (IELC) is responsible for reactively obtaining the routing queries to the destinations outside the cluster. CBF is a hybrid approach, having the potential to be more efficient against traditional schemes in terms of query traffic generation. However, in the absence of appropriate redundant query detection and termination techniques, the CBF may generate more control traffic compared to standard flooding techniques. Interlayer clustering (IELC), composed of an advance query detection and termination technique (QCM), uses link signal strength and a query limit value for detecting flooding and it is capable of minimizing the energy consumption, network flooding, identifying, and eliminating unwanted and redundant flooded queries in IoT networks. A QoS-enabled QCM model is developed and the results of the simulation show the superior performance against state of the art approaches in term of traffic delay, QoS throughput, and energy consumption, under various performance metrics compared with traditional flooding and is state of the art. In order to determine real understanding of flooding in IoT networks we provide modeling of the redundant queries which leads to flooding in Figure 2.

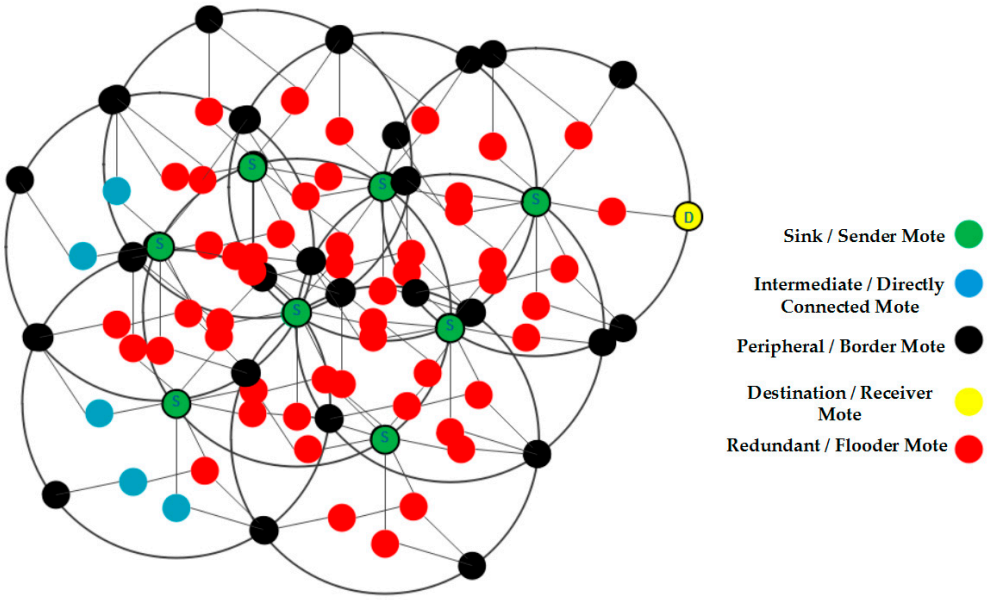

Figure 2. Flooding as a result of redundant queries.

Following is the organized remaining of the paper: Some state of the art related works are outlined in Section 2 on quality of service, energy consumption, searching, flooding, and redundant queries in IoT networks. The network model and problem solution are mainly defined in Section 3 and Section 4. Moreover, Section 4 also explains a cluster based flooding strategy their principles in detail. In Section 5, we design the algorithm for our CBF strategy. Section 6 provides the

experimental results and compares our strategy with the state of the art strategy. Finally, in Section 7 we conclude the paper.

2. Related Works

To address flooding in IoT networks, various defense techniques have been design and developed. Safeguarding the IoT networks from redundant and unwanted flooding has been studied broadly and is not a new problem. As mentioned, DoS attacks flood the network, and may leads to increases in network queries. It also raises network resource utilization, and degrades QoS, along the communication link [15] . In a variety of domains, flooding attacks have been probed, such as a hybrid energy-aware clustered protocol for IoT heterogeneous networks using Hy-IoT proposed by [16]. Hy-IoT provides a real-world cyber IoT architecture based on clusters in managing the heterogeneous IoT network. It also provides an efficient manner of selecting cluster heads which boosts the utilization of the motes energy contents and consequently increases the network life time, as well as the packet transmission rate to the base station. Scalability issues in both the population of motes and the network density towards integrating IoT and SDN controllers is a major challenge in this approach. Ref. [17] proposed an adaptive meta-heuristic search for redundancy in IoT network using the AntClust technique. This approach reduces the cost of the sensor search process and provides an adaptive strategy to maintain the performance of IoT devices against dynamicity in the IoT environment. However, the performance of this approach is only limited to accuracy and no real-time implementation scenario is provided. Ref. [18] used process querying techniques to develop an enabling business intelligence for resource-constrained devices. This approach guides development for process querying methods and specifies a formal instruction to manage the given repository. The lack of good QoS is the major challenge in this approach.

Reference [19] proposed a scalability mechanism for IoT devices. Since scalability has become an important aspect that needs to be considered in any IoT system, the proposed mechanism enables IoT devices to be adaptable to environmental change. The approach lacks QoS and no mechanism to detect redundant queries is provided. Additionally, a three-level framework for IoT redundancy control was proposed by [20]. The framework uses an explicit spatio-temporal data model to control service redundancy at three scales: macro, meso, and micro scales, respectively, in IoT networks. However, many other fundamental components, such as the functional architecture, the algorithms of redundancy data generation and their complexity, as well as the proactive redundancy control scheme of the framework are not be presented. Ref. [21] used a divide-and-conquer (DnC) method to develop an approach for improving energy efficiency in QoS-constrained WSNs (wireless sensor networks). The core idea behind the DnC method is to control the QoS parameters while providing adequate network lifetime. However, there is a need for a real testbed to evaluate the proposed approach in a more realistic manner. Moreover, the overhead of hierarchical MAPE adaptation architecture also needs to be analyzed.

Reference [9] proposed an end-to-end (E2E) QoS specification and monitoring scheme for IoT networks. The authors used service-level agreements (SLA) to determine the specific flooding problems in various IoT devices, such as smart environments, smart cities, smart metering, smart water, smart farming, smart agriculture, industrial control, retail, logistics, domestic, home automation, and e-Health. SLA lacks unified/standard methods for collecting the required metrics across layers and from multiple providers for E2E. Similarly, Ref. [2] proposed a QoS aware resource scheduling for IoT using "Load Balanced Particle Swarm Optimization". The essence of this approach is to reduce application response time and ensures load balancing and provides a minimum response time in an IoT-cloud platform. However, the technique needs to extended to support multiple levels of QoS in IoT networks.

Reference [22] proposed a node-level energy efficiency protocol for IoT devices to improve the energy efficiency in an IoT network. The proposed algorithm uses transmission count makes the decision for finding the shortest hop count. However, the approach does not explore other metrics of QoS. Ref. [23] proposed a QoS architecture for IoT and cloud computing platforms to enable public/users to have easy access over diversified smart applications and services distributed in the

cloud with one IoT-enabled Intelligent Smart Card (ISC) through mobile devices with assured quality of service. The architecture lacks the following metrics of traffic filtering, queue scheduling, congestion management, load balancing, E2E delay management for real-time service, and resource allocation across multiple networks. In addition, modeling QoS in IoT applications was proposed by [24]. The model used a push-out buffer management scheme and preemptive resume (PR) service priority to analyze a finite capacity queue in an IoT network to evaluate the performance of smart devices under varying traffic conditions to ensure preferential treatment of highest priority delay-sensitive data. Long processing time and delays in exchanging information with other devices are major drawbacks in the model. Ref. [25] discussed network architecture and QoS issues in the IoT for a smart city. The authors focused on the communications and networking aspect of the IoT. They identified and proposed a variety of network architectures for smart city applications, and also define their corresponding performance metrics in order to maintain QoS guarantees. However, hardware failure leads to issues of fault tolerance. Ref. [26] proposed a discrete component circuit implementation model together with its computational simulations using Bouali's system. It is an endogenous nonlinear and inseparable cycle that is electronically implemented in circuits. Additionally, a novel countermeasure technique for detecting and curing reactive jamming attacks in IoT networks was proposed in [3]. The proposed technique is called the countermeasure detection and consistency algorithm (CDCA), which is based on threshold values and assumptions to fight reactive jamming attacks.

In accordance with the snags, observed from the current available methods, a QoS-enabled CBF solution is proposed for flooding attacks in IoT networks. The proposed technique is an interoperable solution both for physical layer and network layer devices. CBF divides the whole network into different clusters; local query information is proactively maintained by the IALC, while IELC is responsible for reactively obtaining the routing queries to destinations outside the cluster. CBF is a hybrid approach, having the potential to be more efficient against traditional schemes in terms of query traffic generation. However, in the absence of appropriate redundant query detection and termination techniques, the CBF may generate more control traffic compared to standard flooding techniques. Interlayer clustering (IELC) is composed of advanced query detection and termination techniques (QCM) that link signal strength and QueryLimitThreshold (QLT) values for detecting flooding and it is capable of minimizing the energy consumption, network flooding, and identifying and eliminating unwanted and redundant routing queries in IoT networks. During flooding attacks this technique is accountable for checking mote locality and query detection strength in an IoT network. The strength of query detection is observed for verifying, any variation concerning the signal strength of query packet, and the QLT. The position stability of the mote is based on the location information of the legitimate mote neighbors with its fixed QLT, and observed continuously at different intervals of time.

3. Modeling and Strategy

This section delivers a model for flooding in an IoT network. It elaborates the exchange of query messages in the network between the different motes during interaction, and also indicates the amount of query streams and available motes [20,11]. Furthermore, modeling of the system describes a prime knowledge of the mote functionalities and the flow of queries among end-users and motes [27,28]. Figure 3 illustrates the system model of a sender/sink mote, flooder/redundant mote, and the destination mote. Flooding is solely possible once the flooder mote is physically found between or nearby the target receiver and transmitter motes. When the geometric shape of the system is in such a state, that destination receives the flooder's transmitted signal before it ends transferring or moves on to a new link. Equation (1) represents the different controlled distances D3 and D2 and signifies the fraction of every mote interval that is essential to remain not flooded for successful communications:

$$D2 + D3 \leq (\lambda X_s - X_f)\, c + D1 \quad (1)$$

where mote duration is denoted by X_s, i.e., the total number of legitimate motes in the network, λ denotes the variable allocated to the legitimate motes and it varies between 1 and 10, processing time of the flooder is X_f, i.e., the time taken by the flooding motes to attack the network. $D1$, $D2$, and $D3$ denote the distances between the motes, and c stands for the speed of light, as shown in Figure 3. The flooding mote constantly observes the communication medium, and if a query packet transmission is identified, then straightaway communicates a radio signal for the purpose to induce receiver-side collision [29,30]. The flooder is capable of sending just sufficient energy to debase the received bits of query messages and further leads to failure of cyclic redundancy codes (CRC) checks. Usually, a flooding attack ensures the following criteria: massive efficiency of energy (i.e., low power consumption), small possibility to detect (rather near to 0), attain excessive levels of DoS (i.e., disrupt communications to the desired (or maximum possible) extent) and be resistant to PHY layer anti-flooding techniques (i.e., do not allow signal processing techniques to overcome the attack). Commonly, the criteria of interest are based on the flooding scenario [31,19]. Alternatively, the flooding scenario dictates the appropriate settings for usage [32], such as energy efficacy, will be the prime goal of the malicious motes, because of their small energy resources. According to the aforementioned criterion, a flooder mote tends to maintain the effectiveness in all cases. As such, the flooder mote may adopt some techniques which are stable with MAC layer behaviors to preserve a low possibility of detection.

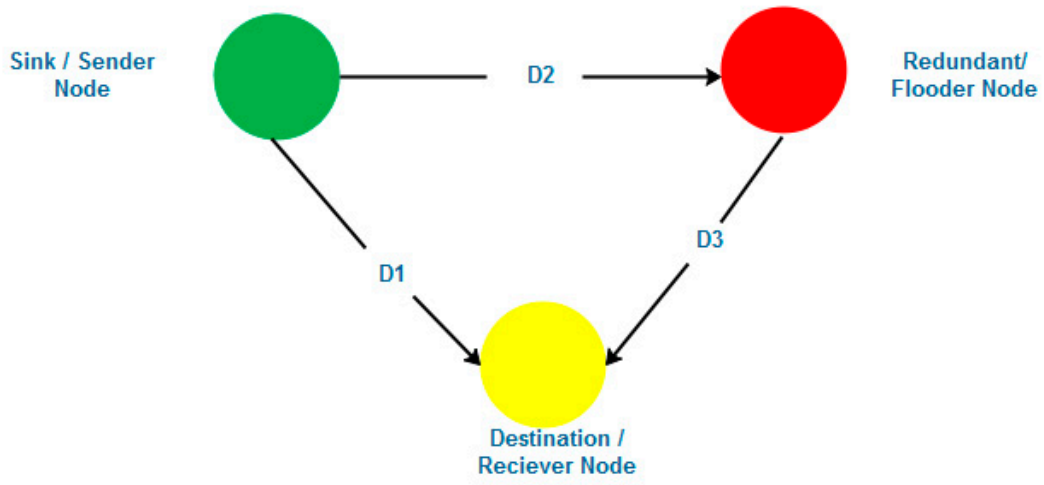

Figure 3. Sink mote, destination mote, and redundant mote system model.

4. Proposed CBF for IoT

Dividing the whole network into different routing clusters is the core idea of CBF. Intralayer clustering (IALC) is responsible for proactive maintenance, with the help of route query exchange and update query packets. The MDP-level MAC initiates route updates, which further updates IALC in the case of broken or established links among the directly-connected neighbor mote, and neighbor-mote is defined as the one which is directly connected and shares a communication link and, thus, is one mote away. The MDP-level (MAC) media access control protocol is responsible for the mote's neighbor's identification, while interlayer clustering (IELC) is responsible to reactively transfer route query packets to motes that reside outside of the mote's cluster via query-reply packets. IELC uses a broadcast delivery service to send the routing queries to its border or peripheral motes. IELC maintains updated route clustering information of peripheral motes using IALC tables; this information is further used to determine that either the query for destination mote refers to their cluster based on QueryLimitThreshold (QLT). QLT assists in regulating the maximum sending ability of motes to send the maximum number of query packets, and also empowers the occurrence of attack detection in the network.

In the following subsection the CBF network, formal definition, algorithm of the MDP, IALC, IELC, QCM, and network assumptions are described.

4.1. CBF Assumptions

A list of the CBF network assumptions [3] are as follows:

- Sensor motes of *n* number are randomly deployed.
- Regarding functionality, all motes ensure the same capability. Every mote equipped with IP address, and can act as a sensor gateway, for the purpose to exchange query messages.
- The mode of communication is single-hop and multi-hop for all connected motes.
- Every mote can initiate a flooding attack. A flooder starts to interrupt with the link, once it detects any activity there.
- The capability of the flooder and normal motes are the same, but the flooder mote is also capable of generating redundant query messages (i.e., random flooding queries).

4.2. Neighbor Mote Discovery Phase (MDP)

In this section a neighbor mote discovery algorithm will look after the maintenance of neighbor and cluster routing tables. Each and every mote maintains neighbor-mote tables and cluster routing tables. The neighbor-mote table along with the neighbor-mote addresses to stores, available QOS parameter values along the link between itself and its neighbor-mote. These parameters are considered for selecting best available routes by the intralayer clustering protocol (used to choose the routes within the cluster). In this phase, each and every mote periodically transmits beacons to its neighbor-motes. On reception of these query packets from a neighbor mote, every mote updates its neighbor table with appropriate values. Each mote exchanges their neighbor tables from their corresponding neighbors and construct cluster routing tables.

Each mote sporadically transmits "Hello" beacons to neighbor-motes, to certify their presence. Once a beacon is delivered by the mote, it directly updates the neighbor-motes table and records the beacon's source. To check the status of mote neighbors, every mote scans the mote tables of their neighbors at periodic sample intervals. The neighbor is considered lost if it beeps no beacons during the Max_previous_list interval samplings. The neighbor is considered found if it beeps beacons. Notification of updated link is sent to IALC, whether a neighbor is found or missing. Figure 4 and Algorithm 1 shows the protocols.

4.3. The IntraLayer Clustering (IALC)

Based on link state information, motes calculate intralayer clustering routes for every extended cluster of motes. Link state updates may receive a mote, either from an interrupt made by the mote discovery protocol (MDP) or from the IALC link state query packet. The link state table keeps an updated list of all link states' information. In addition, the clustering table is recomputed once updates for the waiting link state have been received. Further, links that lie beyond the cluster have been removed and the link state tables are updated. The latest delivered updates of link state and its sources are sent to the mote's neighbor within the cluster. Additionally, a mote shares all intra-cluster link state information to the newly discovered neighbor. The protocol is shown in Figure 5 and Algorithm 2.

4.4. The Interlayer Clustering (IELC)

The core responsibility of the IELC is to discover routes to the hosts that are outside of the mote's cluster. In case a destination mote is not listed in the (IALC) local clustering table, the IELC then triggers a route query request at the network layer. Every route query request is allotted a query-id that is distinctive to the source mote id. The route query request might be uniquely identified by the combination of query-id and source-mote-id in the network.

Once the query-id and source-mote-id have been recorded in the request packet, the query packet is then forward to all the border or peripheral motes of the cluster. Upon receiving the route

query request by the mote, the detected-queries-table records the query-id, source-mote-id, last-hop, and broadcasting-mote. The mote then looks into their routing table to see if the desired destination mote resides inside the cluster. If yes, the mote then responds to the query source and sends back a route query reply, along with a path specified by the last hop information cached in the detected queries table. If the destination mote not reside in the mote's cluster, the mote, then broadcasts the rout query request to all the peripheral motes to search the destination mote outside the cluster. Without the appropriate query control scheme the broadcasting mote, itself, generates more control traffic as compared to flooding. The protocol is shown in Figure 6.

To see the potency of cluster-based querying, a combination of the advanced query control mechanism (query detection and early termination) are employed. Relying on the information stored in the detected queries table, a mote will stop sending route query requests on the departing link if no broadcast receiver mote wants to receive the route query request packet.

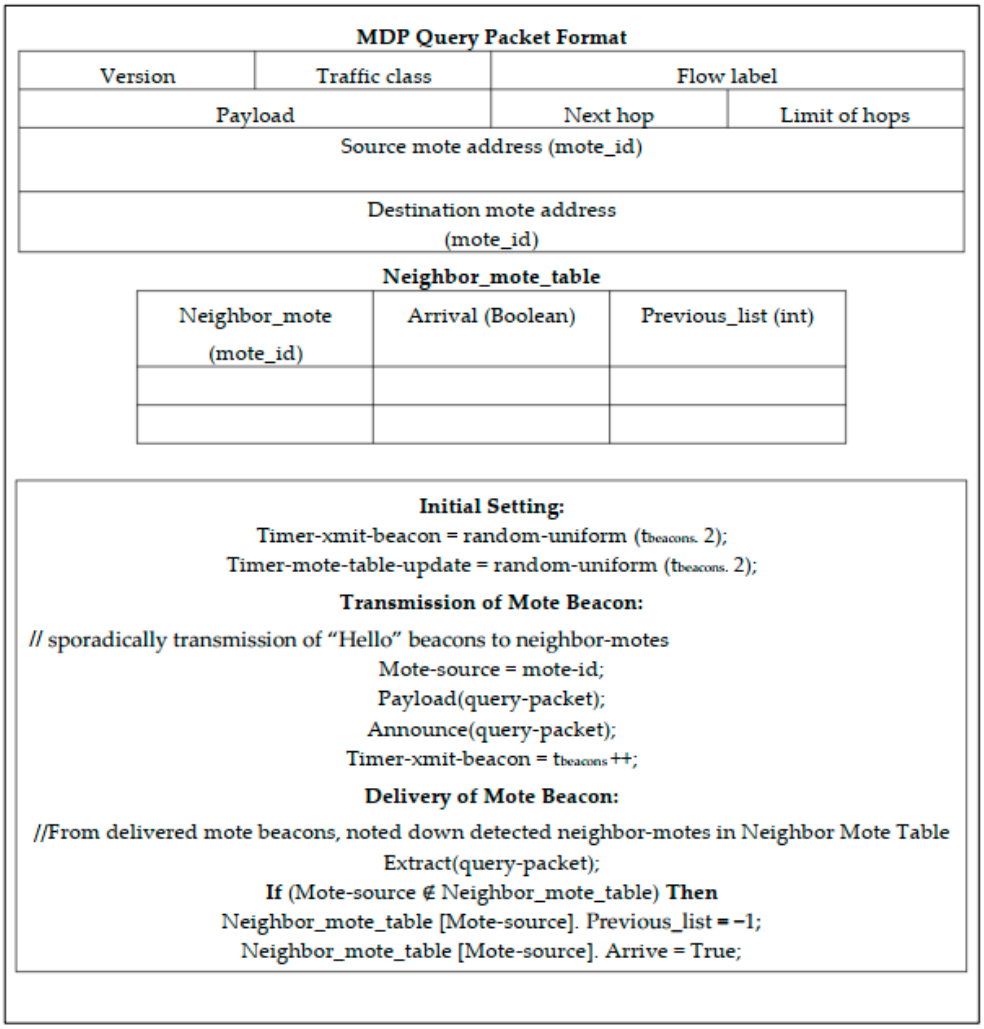

Figure 4. MDP Query Packet format and Neighbor-mote-table.

Algorithm 1 MDP neighbor table update.

//Identify and eliminate the missing and lost neighbors-mote. Every note has neighbor-mote-table.
Begin
Mote-Table
1: **If** (Mote-Table[neighbor-mote]. Arrive == Not True)
2: **If** (Mote-Table[neighbor-mote]. previous_list ≥ Max_previous_list)**Then**
3: Eliminate (Mote-Table[neighbor-mote]);
//If previous_list has not received the neighbor's-mote beacon then eliminate the neighbor-mote

Begin
Mote-Table
1: **If** (Mote-Table[neighbor-mote]. Arrive == Not True)
2: **If** (Mote-Table[neighbor-mote]. previous_list ≥ Max_previous_list)**Then**
3: Eliminate (Mote-Table[neighbor-mote]);
//If previous_list has not received the neighbor's-mote beacon then eliminate the neighbor-mote from the mote-table.
4: Interrupt-load-params (neighbor-mote);
5: Set-interrupt (IALC, "neighbor-mote-lost", "Update IALC Routing Table"); **Endif**
6: **Else** Mote-Table[neighbor-mote]. previous_list ++;
// Increase number of cycles that neighbor-mote's beacon has not been received.
7: **Elseif** (Mote-Table[neighbor-mote]. previous_list == −1) **Then**
// Immediately, alert and update the IALC, if a new neighbor-mote found.
8: Interrupt-load-params (neighbor-mote);
9: Set-interrupt (IALC, "neighbor-mote-found", "Update IALC Routing Table");
10: Mote-Table [neighbor-mote]. previous_list = 0; **Endif**
11: Mote-Table [neighbor-mote]. Arrive = Not True;
12: Update-Timer -Mote-Table = T ++;
13: **End.**

IALC Query Packet Format

Source of the Link (mote-id)	
Destination of the Link (mote-id)	
Source of pk (mote_id)	
Id-link-state (unsigned int)	Reserved

Link State Table

Source of the Link (mote_id)	Destination of the Link (mote_id)	Id-link-state (unsigned int)

Pending Link State Table

Source of pk (mote_id)	Source of the Link (mote-id)	Destination of the Link (mote-id)	Id-link-state (unsigned int)	Status of Link (Boolean)

Cluster Routing Table

Destination (mote_id)	Is_cluster_member (Boolean)	Source routes (mote-id-list)

Figure 5. Intralayer cluster protocol (IALC).

Algorithm 2. Intralayer cluster (IALC) algorithm.

//IALC might be triggered by either from an interrupt made by the mote discovery protocol (MDP) or link state query packet updates.
Begin
1: **If** (query-packet arrived) **Then**
2: Extract(query-packet);
3: My-changed-link = Not True;
4: **Else**
5: Interrupt-extract-params (&destination-link);
6: Source-link = my-mote-id;
7: Source-pk = my-mote-id;

8: Id-link-state = my-id-link-state; **Endif**
9: **Elseif** (Interrupt-type = "Found Neighbor-mote) **Then**
10: Status-link = Up; **Endif**
// Share all intra cluster link states information to the new discovered neighbor-mote.
11: **Elseif (neighbot-mote! (**Link State Table, my-mote-id, link-destination**)) Then**
12: Forward-link-state-table **(**Link State Table, link-destination**);**
13: My-id-link-state ++;
14: **Else**
15: Status-link = Down;
16: **Elseif (neighbot-mote is (**Link State Table, my-mote-id, link-destination**)) Then**
17: My-id-link-state ++;
18: **Endif**
19: **End**

IELC Query Packet Format

Source of the Query (mote-id)			
Destination of the Query (mote-id)			
Last_hop (mote_id)			
Broadcasting-from-mote (mote_id)			
Query-id (unsigned int)	Type-pk (char)	Marker-link (unsigned int)	Hops-maximum (unsigned int)
Route-source [mote 0] (mote_id)			
Route-source [mote 1] (mote_id)			
To			
Route-source [mote N] (mote_id)			

Table for Detected Queries

Source (mote_id)	Query-id (unsigned int)	Last-hop (mote-id-list)	Net-query-coverage

Figure 6. Interlayer cluster protocol (IELC).

Query Control Mechanism

As discussed, the essential aim of the flooder mote is to generate unwanted and redundant routing queries to make the link busy and to stop the authentic mote from sending wanted query packets. A mechanism is required to measure the total time spent for waiting the link to be free, also to check the signal strength of the query packet and location consistency of the mote, and match these metrics to the regular time of the traffic to measure the link for redundant queries. A query control mechanism is introduced to execute these tasks. As shown in Equation (2), the proposed QCM technique uses a change in QueryLimitThreshold (QLT) for detecting and terminating the redundant and unwanted query request packets. The proposed mechanism is favorable in boosting the performance of the IoT's network in terms of signal strength of query packets, and in the improvement in location consistency checking of connected motes, in this way protecting the network against reactive flooding attacks. The QLT is calculated based on Equation (2):

$$\sum_{i=1}^{n} Md_i = P \quad (2)$$

From Equation (2), M is the maximum query request packets of neighbor motes, n denotes the net sum of motes, distances between every mote and QLT are symbolized by d_i (d_i is the distance between the motes that are communicating, and i is the number of respective motes, from 1 to n) and P (P is a value that determines if there is an attack on the network), respectively.

The QCM technique uses link signal strength to determine the consistency of the query packet by scanning if the QLT contradicts the regular packet data transmission value. The mote that transmits the QLT is sporadically tested and matched against the present value to determine the occurrence of flooding in the network. A mote is considered a flooder if its QLT is exceeded the maximum query packet of any mote. Moreover, the mote will not be considered a flooder if its QLT value is smaller than the maximum query packet [33].

Upon detection of a flooder mote by QCM, it alerts all connected neighbor motes to switch their routing paths which lead to the flooder mote. Thereby, QCM ultimately eliminates the flooder mote from the network. The QCM algorithm is shown in Algorithm 3.

Algorithm 3. Query control mechanism (QCM) algorithm.

Begin
n, MaxQuery(M): M ∈ Neighbor motes
Input: query limit value, x_0, y_0, x_n, y_n;
Output: Dist, Δ query limit value;
1: **If** (MaxQuery(M) < QueryLimitThreshold(QLT)) **Then**
2: Check (SignalStrengthConsistancy(SSC));
3: Check (Sending QueryLimitThreshold (QLT));
4: Check-link = (SignalStrengthConsistancy(SSC), MaxQuery(M));
5: **Elseif** (Check-link = Not True) **Then**
6: Flooding occurred; **Endif**
9: **If** (MaxQuery(M) > QueryLimitThreshold (QLT)) **Then**
10: $Z_0 = (x_0, y_0)$ = mote_location;
11: $Z_n = (x_n, y_n)$ = find_mote_location;
12: Check-link = Query Packet Sent;
13: **Elseif** (Check-link = Not True) **Then**
14: Flooding occurred; **Endif**
15: **End.**

n exhibits the sum of motes within the network, Z_0 denotes the initial mote position (x_0, y_0), Z_n denotes the current location of the mote (x_n, y_n), and M symbolizes neighbor motes.

The QCM algorithm is composed of two levels. The initial level is responsible for selecting the information that sends the QueryLimitThreshold (QLT) and checks the SignalStrengthConsistancy (SSC) for each mote. As all motes have the ability to send information after a specific duration, the mechanism can record the volume of queries sent by every mote. The level is to check the SignalStrengthConsistancy (SSC), that is depending on the sending QueryLimitThreshold (QLT).

It is assumed that each mote in the network might be in one of the following three conditions: typical (i.e., the mote is in a normal condition), fishy (i.e., the mote is in a suspicious condition), and the flooder (i.e., the mote is a flooder). At the beginning, all motes are in the typical condition and can exchange query packets to each other by single- or multi-hop communication. In the fishy condition path analysis is carried out based on the communication type (single- or multi-hop) which is further utilized by the motes to query data exchange. If the fishy sender mote is based on single-hop communication, the single hop path analysis can be done to easily detect the fishy sender mote. If the fishy mote is based on multi hop communication, the path analysis can be done to check every hop and query the packets transmitted by all motes. A mote is considered a flooder if normally sent queries contradict with the number of query packets transmitted by the mote. The mechanism considers a mote is in the typical condition if the amount of MaxPacket (M) is similar to the QueryLimitThreshold (QLT). Lastly, the proposed QCM eliminates all the flooder motes by updating all connected neighbor motes to change their communication channels and links that come from the flooder mote. Figure 7 elaborates the overall process flow chart of cluster-based flooding.

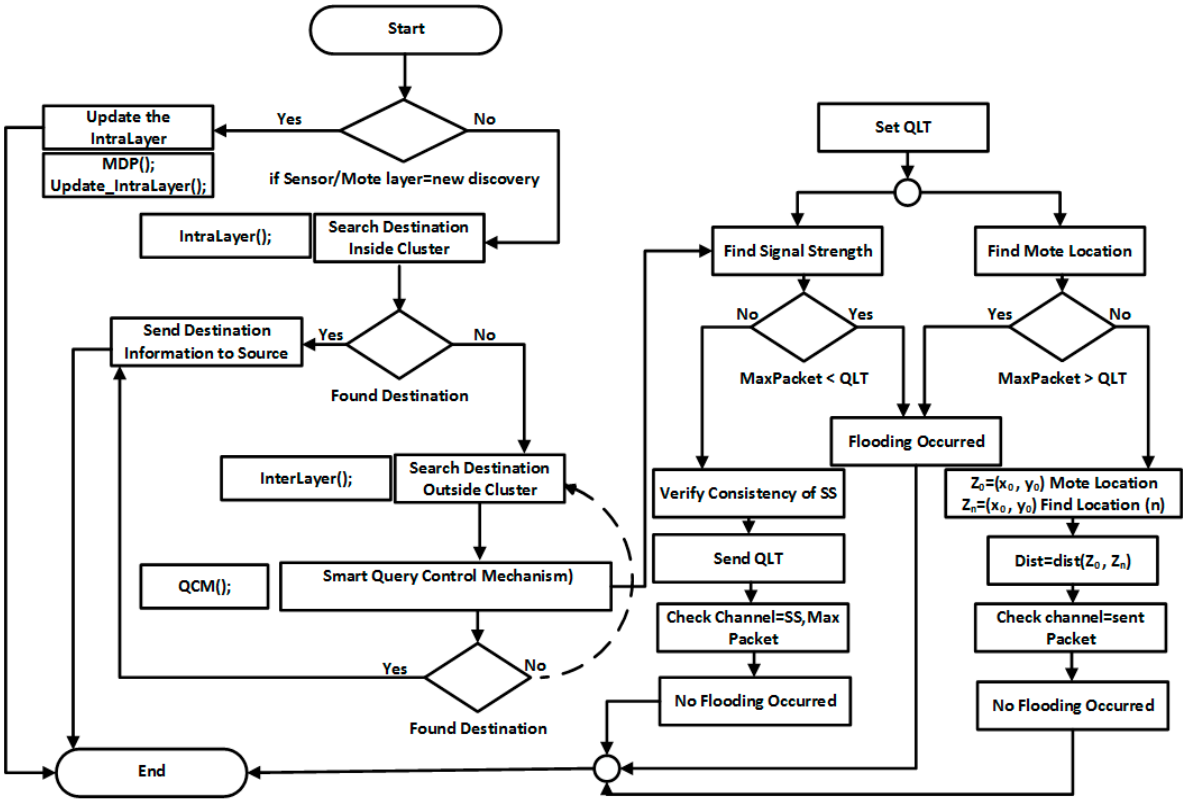

Figure 7. Flowchart of the CBF process.

4.5. Cluster Based Flooding (CBF) Model Formulation

The cluster-based flooding model is designed as either one of two sorts of games: proactively (intralayer) favorable for the sensor layer or reactively (interlayer) favorable for the network layer. The aim of intralayer clustering is to use more network resources to get the updated information of the motes for handling priority packet queries and mitigate delays in the IoT network by using the table-driven approach. The cluster-based flooding (CBF) model is subsequently elaborate. Assuming that during query propagation the topology of the network remains static, and intralayer clustering (IALC) is already aware of topology variation. Intralayer clustering (IALC) ultimately enhances CBF by query route repairing and caching inside the cluster. A significant feature of this method is that it is not dependent on assumptions concerning the size of any mote's cluster. Subsequently, this proof of correctness refers to the networks where every mote has the same cluster radius, and to the networks where every mote individually handles their own cluster.

As shown in Figure 8, let m(t) be the set of queried motes which reside within the cluster (IALC), which might be interior or peripheral motes at time interval t. We refer to queried motes as, i.e., motes that have already been visited and are directly accessible to the source mote. Similarly, the remaining set of unqueried motes are denoted by m^c (t), unqueried motes are referred to the motes that are not yet visited, reside in the outer cluster, and are not directly accessible to the source mote. Let B(t) be the subset of m(t), which is referred as peripheral or border motes. Every peripheral mote B(t) holds almost one neighbor in unqueried motes m^c (t) (B(t)→ m^C(t)), peripheral motes are the covered motes that construct a boundary between queried and unqueried clusters in the network, as listed in Equation (3).

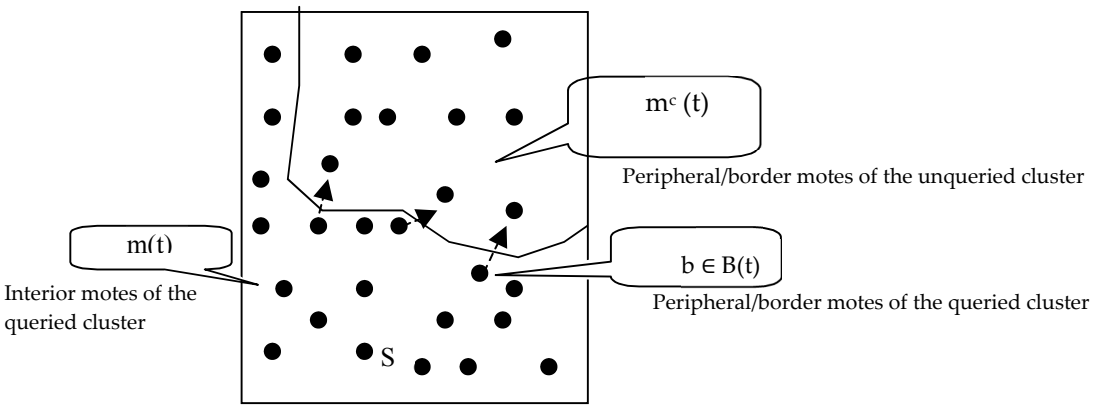

Figure 8. Cluster-based flooding (CBF) model formulation.

$$m(t_1) \subset m(t_2) \text{ and } m^C(t_2) \subset m^C(t_1), \text{ for } t_1 \leq t_2 \quad (3)$$

In Equation (3), once a mote has been queried, it cannot be unqueried or uncovered. Therefore, $m(t_1) \subset m(t_2)$.

From the fundamental set principle, it also follows that $m^C(t_2) \subset m^C(t_1)$, as shown in Equation (4):

$$|m^C(t)| > 0, \ |B(t)| > 0 \quad (4)$$

From Equation (4), since all motes that belong to uncoverd mote $m \in m^C(t)$ are accessible from other motes, which has a covered peripheral neighbor $b \in B(t)$ (b signify the set of queried motes which resides on the cluster boundary). Hence peripheral mote b was visited by a query, Therefore, b either be an interior or a peripheral mote of the source mote. If b is an interior mote, then every b's neighbor means to be queried as well. Though, even some of b's neighbors are unqueried which we referred as uncovered motes $m \in m^C(t)$. Therefore, b needs to be the border or a peripheral mote of a source mote as shown in Equation (5) as:

$$b \in B(t_1): b \notin B(t_2); \ |Mc(t_2)| < |Mc(t_1)| \quad (4)$$

From Equation (5), assume that N_b is symbolized as the set of motes which are the neighbors of mote b. Then, the following two conditions will occur at p:

1. If $(b \in B(t_1)$, then $m^C(t_1) \cap N_b \neq \emptyset$
2. If $(b \notin B(t_2)$, then $m^C(t_2) \cap N_b \neq \emptyset$

The above conditions simply follows that $m^C(t_1) \neq m^C(t_2)$, we have also shown earlier that $m^C(t_2) \subset m^C(t_1)$, so $|m^C(t_2)| < |m^C(t_1)|$.

If a mote b is a broadcast receiver of a source mote and obtains the route query updates at interval t_2 then $b \notin B(t_2)$. That is, when mote b acquires the route query updates at interval t_2, it starts searching the queried destination mote, thus updating all its associated motes of their cluster. Therefore, all mote b's neighbors might be covered and $b \notin B(t_2)$.

The second type (interlayer/reactive) is a sort of two player game, in this game one player 1 acts as a "maximizer" and player 2 acts as a "minimizer". The aim of the maximizer (player 1) is to attain utmost volume of energy gains, while the minimizer (player 2) strives to retain the least volume of energy gains [34]. Selection of this approach is because of, every mote (player) in the IoT network have the tendency to make the maximum use of resources referred as network gains during transmission of the route query packets. These players act as an observer mote and have the responsibility to detect the unwanted flooding in the network. The players are denoted as X1 and X2, where X1 stands for the inspecting mote and X2 denotes the flooder mote.

$$X = \{ X1, X2\} \quad (6)$$

The back and forth in the cluster-based flooding (CBF) model designing are *(Ic)* constant inspecting and *(Ip)* periodic inspecting, which permit the communicating motes to inspect the link

either constantly or at a preset interval of time. The adopted reactive flooding strategy is symbolized as "ReFa". To inspect the communication link, inspecting mote uses the *(Ic, Ip)* strategies, denoted as:

$$X = X1 * X2 \quad (7)$$

$$X1 = \{Ic, Ip\} \quad (7a)$$

$$X2 = \{ReFa\} \quad (7b)$$

where *X*1 and *X*2 are player 1 and 2, respectively.

To indicate the inspecting mote efficacy, here we consider the player utility function, to verify that either the flooding attacks are detected effectively or not. The false positive and detection rate is considered as two utility functions for the inspecting mote. The aim of the flooder utility is to stop the successful transmission of route query packets and decrease the network throughput and QoS by attempting unwanted and redundant query strikes. The utility function (F_u) is set as follows, where detection rate and flooding attack gains are denoted as F_u 1 and F_u 2, respectively:

$$\{F_u\} = \{F_u1, F_u2\} \quad (8)$$

A reactive flooding attack strategy is formulated based on Equations (2), (6), and (7) for *Ic* and *Ip* shown below. Where F_d represents the duration of the flooding attack, F_{dg} stands for the gain of flooding detection, t denotes the I_p interval of time, F_{ag} symbolizes for the gain of a successfully launched flooding attack, the payoff of the reactive flooding is denoted by P_{refa}, and both (P_c, P_p) are the payoffs when (*Ic*, *Ip*) are used to sense the flooding attack. Payoff refers to the cost of initiating or detecting an attack:

$$Ic = F_d(F_{ag} - P_{refa}), (F_{dg} - P_c) \quad (9)$$

$$Ip = F_d(tF_{ag} - P_{refa}), t(F_dF_{dg} - P_p) \quad (10)$$

The strategy for the reactive flooding attack is expressed in Equation (7)–(7b) for both (*Ic*, *Ip*). The flooder mote will immediately trigger an attack if it detects any activity in the communication link. All mathematical symbols used in Sections 3 and 4 are defined in Table A1 of the Nomenclature A.

5. Implementation Facts

This section elaborates the implementation details of the proposed approach. A cutting edge simulator known as Contiki Cooja is used to form redundant flooding scenarios [35] under realistic scenarios with a number of malicious motes and different intervals of traffic. A detailed specification of the simulation parameters are listed in Figure 4, in accordance with the IEEE 802.15.4 radio regulation. We used random topology because of the heterogeneous nature of IoT devices to present connectivities of sensor mote capabilities. The network size of the simulation model is 1–16 motes and are randomly deployed in an area of 100 m^2, where all motes are active transmitters and receivers. Furthermore, we evaluate the simulation based on three scenarios (i.e., varied intervals of traffic, varied number of malicious motes, and realistic scenarios) in all cases the inner arrival timing is exponential. Regarding traffic intensity, we explore different levels of saturation in the network by varying the traffic intensity, which may vary from low saturated networks (p_{max} = 0.1) up to highly saturated networks (p_{max} > 1). Values mentioned in Table 1 provide the foundation for an appropriate assessment of reactive flooding [36,37] .

Table 1. Simulation parameters.

Mote Type	Sky Mote
Energy at initial state	100 J
Power at idle state	31 mW
Power at receiving state	35 mW
Power at sending state	31 mW
Power at sleep state	15 μW

Simulation name	QoS Enabled CBF
Radio medium	UDGM with Distance Loss
Startup Delay	1 ms
Random Seed	123,456 (Default)
Positioning	Random
Topology	Random
Number of Motes	16

The radio medium used for simulation is the Unit Disk Graph Medium (UDGM: distance loss) having a transmission range of 50 m and interference range of 100 m with an initial energy of 100 joules(J). Each device moves randomly with startup delay of one millisecond (ms). Each node consumes 31 milliwatt(mW) energy in an ideal state and 15 microwatt (μW) in the sleeping state. A total of 35 and 31 mW energy is consumed during data receiving and sending states, respectively. Moreover, the transmission delay for high-speed links is insignificant. For example, a 1500-byte packet transmitted over a 155 Mbps STM-1/OC-3 link would take only 0.08 ms.

6. Results and Discussion

The performance estimation and evaluation of the proposed technique against up-to-date DnC, SLA [2,3], and Hy-IoT [32] methods for tracing and mitigating the unwanted and redundant reactive flooding are described in this section. The routing protocol and MDP protocol [24] are ad hoc routing and Contiki, respectively. To obtain the appropriate results simulations are performed 60 times based on the following three scenarios:

- Scenario based on varied intervals of traffic: This condition plays an important role to gauge and ensure the effectiveness of flooding attacks and to regulate the defensive techniques in varying interval of traffic. Ranges for traffic interval is set as 1–10 s, where 1 s is referred to as faster and 10 s is slower.
- Scenario based on a varied number of mischievous motes: this condition is favorable in analyzing the impact of flooding attack on the network and to take the appropriate action to counter mischievous motes. Motes 2, 6, 10, 15 are set as mischievous motes, and the interval of traffic is set to 1 s, where 1 s is referred as the fastest traffic in the network.
- Condition based on realistic scenario: In this conditional scenario motes are restricted to not transfer the route query information simultaneously; they are only allowed to transfer route query requests at different intervals of time. These intervals are randomly set from 1–10 s.

6.1. Average Consumption of Energy

Figure 9 exhibits the average consumption of energy and number of malicious motes at varying intervals of time. In Figure 9a the proposed QCM technique outperformed compared to DnC, SLA, and Hy-IoT approaches in terms of dropping the average consumption of energy. Since the proposed technique is capable of detecting flooder motes and detach them from the network, in this way it reduces the level of energy consumption that arises during redundant and unwanted flooding attacks, whereas the average energy consumption of DnC and SLA is approximately 21 and 18%, respectively, from 1–5 s intervals, and this ratio continuously rises as the interval increases. However, in the case of the proposed mechanism the ratio of consumption of energy falls to 6% as compared to the 13% of the existing Hy-IoT approach.

Obviously, for all techniques the average energy consumption is directly proportional to the time interval, as the time interval increases average energy consumption also increases. The flooder mote consumes the highest level of energy as it directly sends redundant queries once detect any communication activity in the network. Figure 9b demonstrates the average consumption of energy along with different numbers of mischievous motes, mischievous motes are resides at 2, 6, 10, and 15.

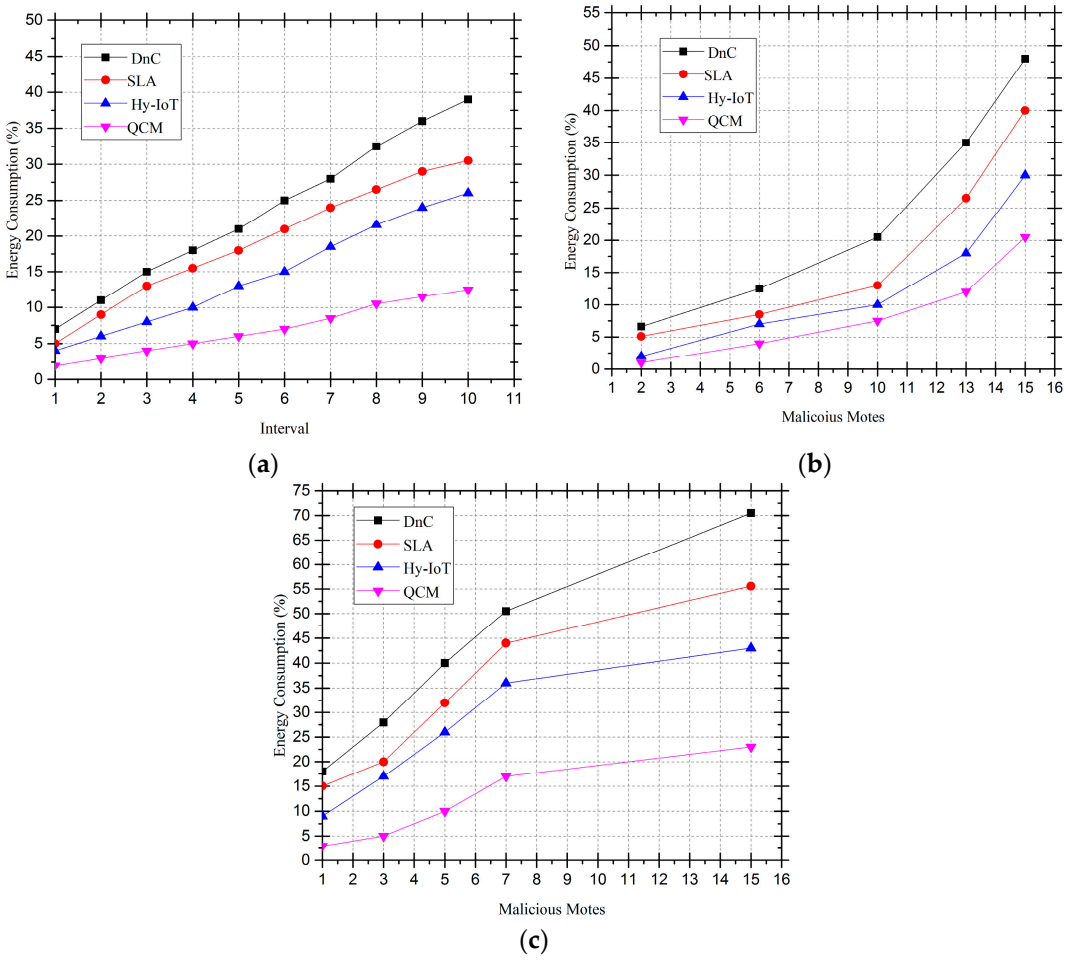

Figure 9. Average consumption of energy with respect to different interval of traffic (**a**); with malicious motes (**b**), and malicious motes with realistic condition (**c**).

An exact realistic analysis of QCM is conducted to find the level of mischievous motes during flooding expansion in the network. It is evident from the result that in the presence of malicious motes the level of energy consumption increases gradually. At malicious mote 2 the levels of energy consumption are approximately 8 and 5% for DnC and SLA approaches, respectively, and at malicious mote 15 this consumption level reaches to approximately 48% and 40%. Hence, by introducing QCM this level falls to approximately 2%, 4%, and 20% at malicious motes 2, 6, and 15, respectively. The result of proposed QCM is superior then the current Hy-IoT, that is about 3% and 8% at malicious motes 2 and 6, respectively, and subsequently moves upward to 32% at mote 15. The proposed QCM is capable of reducing the level of energy consumption by suspending the flooder mote from redundant transmission to the active state, and treat the flooding attacks effectively and efficiently.

Figure 9c demonstrates the average consumption of energy along with malicious motes and realistic conditions. A realistic condition is referred to as the random transmission of route query packets between the motes at varying intervals. During the redundant flooding attack, it is very essential to simulate the realistic condition to understand the efficiency of the proposed QCM. As shown in the result the average consumption of energy of DnC and SLA at malicious mote 1 is approximately 18% and 15%, respectively, and this consumption rises to about 70% and 56%, respectively, at malicious mote 15, while with QCM this consumption of energy falls to approximately 3% and 24% at malicious motes 1 and 15, respectively. In addition, this consumption is approximately 6% less than that of the traditional Hy-IoT at malicious mote 1 and 20% at malicious mote 15 using the same condition. The performance of the network improves by adopting the proposed QCM Technique.

6.2. Traffic Delay

The effect of traffic delay on number of malicious motes, the time interval and malicious motes in realistic condition are shown in Figure 10a–c, respectively. In Figure 10a, QCM outperformed as compared to DnC, SLA, and Hy-IoT by having least traffic delays. QCM has the ability to detect, pause, and detach the flooding mote from the network, which helped in improving its performance. On the other hand, the redundant and unwanted queries were also removed by detaching the flooding motes. Traffic delay of DnC and SLA is 26% and 20%, respectively at Interval 1 as shown in Figure 10a. The proposed algorithm in the network causes a decrease of 10%, approximately in the traffic delay at the same interval. This drop in traffic delay by the proposed algorithm is approximately 4% less than the traditional Hy-IoT. Hence, this percentage is directly proportional to the interval: as the interval increases the percentage will continue to increase.

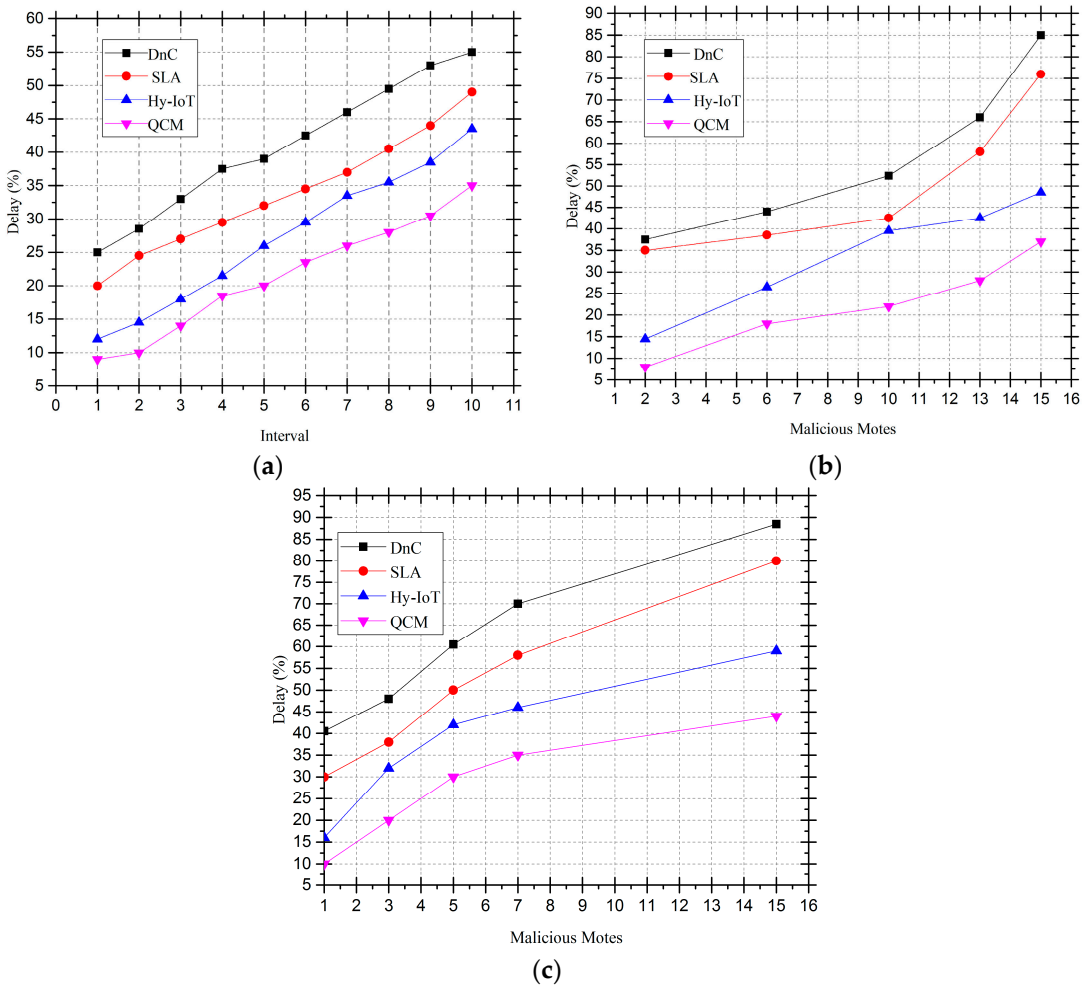

Figure 10. Delay with respect to different interval of traffic (**a**), with malicious motes (**b**), and malicious motes with realistic condition (**c**).

The traffic delay with respect to the increasing number of malicious motes are shown in Figure 10b. Here, at malicious mote 2, the traffic delay of DnC and SLA are about 37% and 35%, respectively, in normal reactive flooding. This increase in traffic delay continues to increase in the number of malicious motes. At malicious mote 15 the maximum traffic delay of DnC and SLA are recorded as approximately 85% and 75%, respectively. The proposed QCM outperformed as compared to Hy-IoT at malicious 2 and 15, and it reduces the delay to about 7% and 37% gradually, while with the same malicious motes (2 and 15) the Hy-IoT has a comparatively higher traffic delay of approximately 15% and 49%, respectively. This decrease in traffic delay by QCM is due to the decrease in the link waiting time.

The proposed flooding techniques are also evaluated using realistic network conditions with an increasing number of malicious motes as mention in Figure 10c. Traffic delay of DnC and SLA at malicious mote 1 is approximately 45% and 30%, respectively. This traffic delay of DnC and SLA at malicious mote 15 increases up to approximately 89% and 81%, respectively, in the realistic network scenario. The proposed QCM comparatively outperformed in the realistic network scenario by having approximately 11% and 45% traffic delays at malicious motes 1 and 15, respectively. This drop in traffic delay by QCM is approximately 6% and 15% better than the existing Hy-IoT at motes 1 and 15, respectively.

6.3. QoS and Throughput

As shown in Figure 11, the proposed QCM is compared with DnC, SLA, and Hy-IoT using network throughput, which we referred to as QoS. Here QoS is measured for these four flooding mechanisms using three scenarios, such as time interval, increasing number of malicious motes and malicious motes with realistic network conditions. In Figure 11a, network throughput (QoS) of DnC and SLA are decreases up to approximately (44% and 61%), respectively at an interval of 1 s. While, with the proposed QCM the QoS boosts to approximately 85%, this result is 9% better as compared to existing Hy-IoT at the same time interval 1 s. According to the result as shown, the performance of network throughput (QoS) is inversely proportional to the time interval, as QoS decreases with the increase in the interval. Flooder motes keep engaged the transmission link, which generates unwanted and redundant routing request queries and directly affects the QoS of the IoT networks.

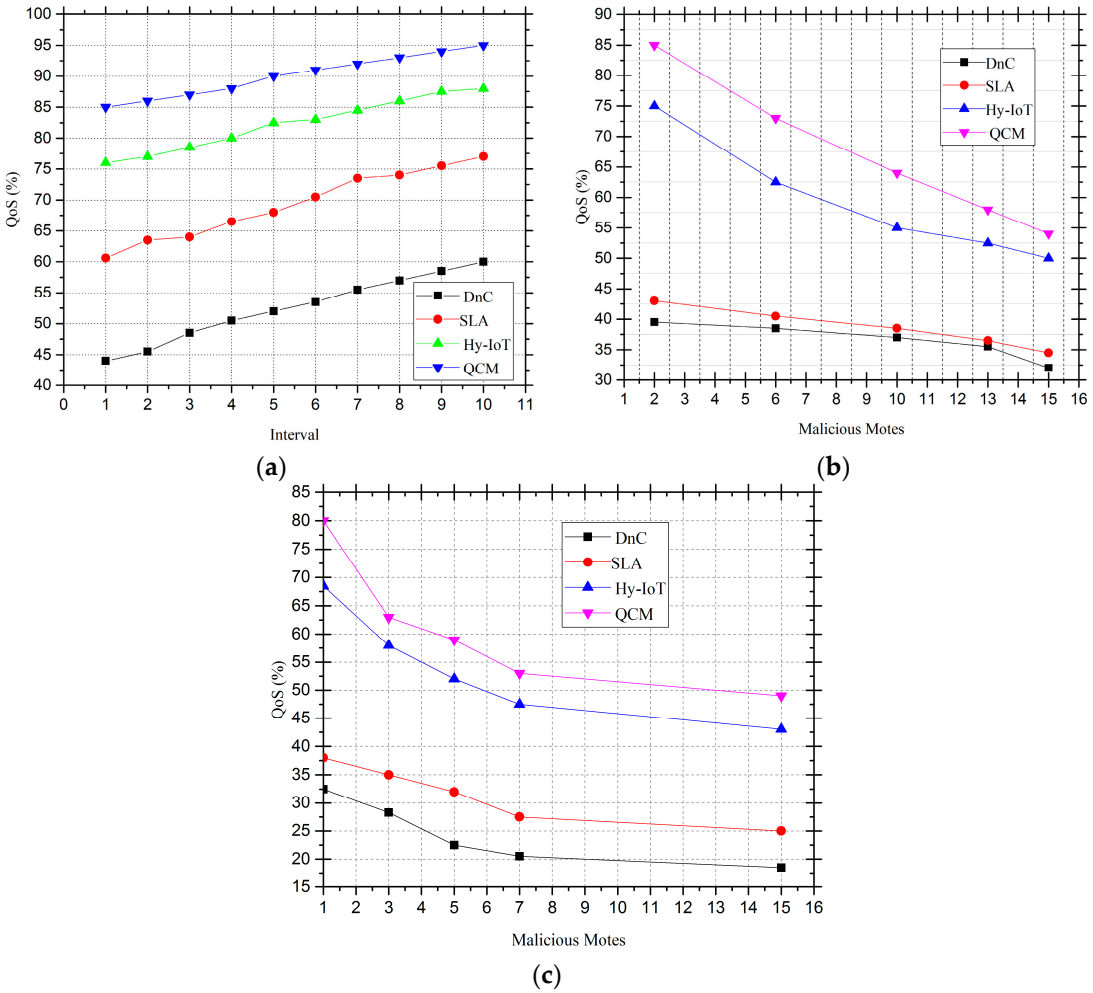

Figure 11. QoS with respect to different interval of traffic (**a**), with malicious motes (**b**), and malicious motes with realistic condition (**c**).

Figure 11b shows the network throughput (QoS) of QCM, DnC, SLA, and Hy-IoT with increase in the number of malicious motes. Here, initially at malicious mote 2 the network throughput (QoS) of DnC and SLA is approximately (39% and 44%), and finally at malicious mote 15 this percentage gradually drops to approximately (30 and 35%), respectively. While with proposed QCM the (QoS) at malicious mote 2 is about 86% and gradually drops to approximately 53% at malicious mote 15. This results are approximately (9% and 6%) better as compared to the existing Hy-IoT at malicious motes (2 and 15), respectively. It is because; the proposed mechanism is capable of providing fast link access to the motes during redundant query attacks.

Network throughput (QoS) of QCM, DnC, SLA and Hy-IoT is analyzed in realistic network conditions with increasing number of malicious motes as shown in Figure 11c. The simulation results revealed that the network throughput of DnC and SLA with respect to malicious mote 1 is approximately (34% and 37%), respectively, and this ratio drops to approximately (17% and 25%), respectively at malicious mote 15. The proposed QCM mechanism boosts the QoS throughput to approximately (80% and 49%) at malicious motes (1 and 15), respectively. This increase in result of network throughput by QCM is approximately (12% and 5%) better as compared to the existing Hy-IoT at malicious motes (1 and 15), respectively in a realistic condition.

7. Conclusions and Future Work

This paper has studied numerous defensive techniques against unwanted and redundant routing queries which may lead to heavy network traffic and flooding in IoT networks. In this study, we implement the reactive part Interlayer clustering (IELC) of cluster-based flooding (CBF), and proposed a query control mechanism (QCM) to detect and terminate the unwanted and redundant queries which is based on link signal strength, consistency of query packets, and a query limit threshold. Furthermore, it is evident from the results that proposed QCM appeared superior performance compared with the state of the art defensive techniques in terms of average consumption of energy, traffic delay, and QoS, which we referred to as network throughput. Thus, QCM drops the average consumption of energy to almost 6% compared to the DnC, SLA, and Hy-IoT that consume approximately (21%, 18%, and 13%) under varying intervals of traffic. The performance of QCM is also better regarding average consumption of energy with malicious motes against the traditional approaches by dropping the consumption at motes (2, 6, and 15), respectively to about (2%, 4%, and 20%). Additionally, QCM also exhibits dominant performance regarding network delay by decreasing the delay to about 10%, which is 4% less as compared to the state of the art. While in cases of malicious motes, the proposed QCM drops the network delay to approximately (7% and 37%) at malicious motes (2 and 15), respectively. Lastly, QCM enhances the amount of QoS to 85%, that is 9% greater as compared to Hy-IoT. The proposed QCM technique employs QueryLimitThreshold (QLT) for detecting and terminating the redundant and unwanted query request packets, and in this way boost the IoT network performance in term of signal strength of query packet and improved the location consistency checking of connected motes, to keep the network away from reactive flooding attacks. This performance clearly shows the difference between our approach and the contemporary approaches. We have planned to extend this work in the future by considering discrete component circuit implementation model using Bouali's system to detect some other attacks in IoT by extending the number and types of motes in order to test the reliability of our approach in the presence of many motes. Additionally, we have planned to include the proactive part, intralayer clustering (IALC) of the CBF, which is favorable in high priority and lower delay IoT networks, i.e., smart transportation, smart health, and smart security, and to model a physical prototype for it.

Author Contributions: As a PhD candidate, F. A.K. proposed and implemented the main idea under the supervision of R.M.N. and M.L.M.K. N.M.N. organized the flow of the manuscript. S.M.A. and A.U.R. refined the manuscript and responses to the reviews.

Funding: This research work was supported by Faculty Grant GPF005D-2018 (University of Malaya).

Conflicts of Interest: All authors declare that they have no conflict of interest.

Appendix A

Table A1. Mathematical list of symbols.

X_s	Duration of the mote
X_f	Processing time flooder mote
c	Speed of light
D1, D2, and D3	Distances
M	Maximum query request packets of neighbors
x	Net sum of motes
d_i	Distances between every mote
P	QueryLimitThreshold (QLT)
Z_0	Mote position at the beginning from (x_0 to y_0)
Z_n	Mote position at the end from (x_n to y_n)
m(t)	Set of queried motes
$m^c(t)$	Set of unqueried motes
B(t)	Set of peripheral or border motes
t, t_1, and t_2	Different time intervals
X1	Inspecting mote
X2	Flooder mote
I_c	Constant inspecting
I_p	Periodic inspecting
F_u	Utility function
F_d	Duration of the flooding attack
F_{dg}	Gain of flooding detection
P_{refa}	Payoff of the flooding
P_c, P_p	Payoffs for (Ic, Ip)

References

1. David, D.R.; Nait-Sidi-moh, A.; Durand, D.; Fortin, J. Using Internet of Things technologies for a collaborative supply chain: Application to tracking of pallets and containers. *Procedia Comput. Sci.* **2015**, *56*, 550–557, doi:10.1016/j.procs.2015.07.251.
2. Krishnapriya, S.; Joby, P.P. QoS Aware Resource Scheduling in Internet of Things-Cloud Environment. *Int. J. Sci. Eng. Res.* **2015**, *6*, 294–297.
3. Fadele, A.A.; Othman, M.; Abaker, I.; Hashem, T.; Yaqoob, I.; Imran, M.; Shoaib, M. A novel countermeasure technique for reactive jamming attack in internet of things. *Multimed. Tools Appl.* 2018, doi:10.1007/s11042-018-6684-z.
4. Fadele, A.A.; Othman, M.; Hashem, I.A.T.; Alotaibi, F. Internet of things Security: A Survey. *J. Netw. Comput. Appl.* **2017**, *88*, 10–28, doi:10.1016/j.jnca.2017.04.002.
5. Laxmi, P.; Deepthi, G.L. Smart Water Management Process Architecture with IoT Based Reference. *Int. J. Comput. Sci. Mob. Comput.* **2017**, *6*, 271–276.
6. Yan-E, D. Design of intelligent agriculture management information system based on IoT. In Proceedings of the 4th International Conference on Intelligent Computation Technology and Automation, ICICTA 2011, Shenzhen, China, 28–29 March 2011; Volume 1, pp. 1045–1049, doi:10.1109/ICICTA.2011.262.
7. Huang, J.; Duan, Q.; Zhao, Y.; Zheng, Z.; Wang, W. Multicast Routing for Multimedia Communications in the Internet of Things. *IEEE Internet Things J.* **2017**, *4*, 215–224, doi:10.1109/JIOT.2016.2642643.
8. Dhumane, A.; Prasad, R.; Prasad, J. Routing Issues in Internet of Things: A Survey. In Proceedings of the International MultiConference of Engineers and Computer Scientists 2016 Vol I, IMECS 2016, Hong Kong, China, 16–18 March 2016.

9. Alqahtani, A.; Solaiman, E.; Buyya, R.; Ranjan, R. End-to-End QoS Specification and Monitoring in the Internet of Things. 3–6. 2016. Available online: https://pdfs.semanticscholar.org/6156/3ae040aef11fd7dded6d39d92516c0368423.pdf (accessed on 22 December 2018).
10. White, G.; Nallur, V.; Clarke, S. Quality of service approaches in IoT: A systematic mapping. *J. Syst. Softw.* **2017**, *132*, 186–203, doi:10.1016/j.jss.2017.05.125.
11. Rizal, R.; Riadi, I.; Prayudi, Y. Network Forensics for Detecting Flooding Attack on Internet of Things (IoT) Device. *Int. J. Cyber-Secur. Dig. Forensics (IJCSDF)* **2018**, *7*, 382–390.
12. Liang, O.; Ahmet Şekercioğlu, Y.; Mani, N. A low-cost flooding algorithm for wireless sensor networks. In Proceedings of the IEEE Wireless Communications and Networking Conference, WCNC, Kowloon, China, 11-15 March 2007; pp. 3498–3503, doi:10.1109/WCNC.2007.641.
13. Raju, I.; Parwekar, P. Detection of sinkhole attack in wireless sensor network. *Adv. Intell. Syst. Comput.* **2013**, *381*, 629–636, doi:10.1007/978-81-322-2526-3_65.
14. Arkian, H.R.; Atani, R.E.; Pourkhalili, A.; Kamali, S. A stable clustering scheme based on adaptive multiple metric in vehicular Ad-hoc Networks. *J. Inf. Sci. Eng.* **2015**, *31*, 361–386, doi:10.1002/dac.
15. Kharkongor, C.; Chithralekha, T.; Varghese, R. A SDN Controller with Energy Efficient Routing in the Internet of Things (IoT). *Procedia Comput. Sci.* **2016**, *89*, 218–227, doi:10.1016/j.procs.2016.06.048.
16. Sadek, R.A. Hybrid energy aware clustered protocol for IoT heterogeneous network. *Future Comput. Inform. J.* 2018, 1–12, doi:10.1016/j.fcij.2018.02.003.
17. Ebrahimi, M.; ShafieiBavani, E.; Wong, R.K.; Fong, S.; Fiaidhi, J. An adaptive meta-heuristic search for the internet of things. *Future Gener. Comput. Syst.* **2017**, *76*, 486–494, doi:10.1016/j.future.2015.12.006.
18. Polyvyanyy, A.; Ouyang, C.; Barros, A.; van der Aalst, W.M.P. Process querying: Enabling business intelligence through query-based process analytics. *Decis. Support Syst.* **2017**, *100*, 41–56, doi:10.1016/j.dss.2017.04.011.
19. Gupta, A.; Christie, R.; Manjula, P.R. Scalability in Internet of Things : Features , Techniques and Research Challenges. *Int. J. Comput. Intell. Res.* **2017**, *13*, 1617–1627.
20. Haddad, H.; Bouyahia, Z.; Jabeur, N. Towards a Three-Level Framework for IoT Redundancy Control through an Explicit Spatio-Temporal Data Model. *Procedia Comput. Sci.* **2017**, *109*, 664–671, doi:10.1016/j.procs.2017.05.373.
21. Abdelaal, M.; Theel, O.; Kuka, C.; Zhang, P.; Gao, Y.; Bashlovkina, V.; Nicklas, D.; Fränzle, M. Improving Energy Efficiency in QoS-Constrained Wireless Sensor Networks. *Int. J. Distrib. Sens. Netw.* 2016, doi:10.1155/2016/1576038.
22. Vellanki, M.; Kandukuri, S.P.R.; Razaque, A**.** Node Level Energy Efficiency Protocol for Internet of Things. *J. Theor. Comput. Sci.* **2015**, *3*, 1–5, doi:10.4172/2376-130X.1000140.
23. Premila, D.; Rabara, A.; Jerald, V. Quality of Service Architecture for Internet of Things and Cloud Computing. *Int. J. Comput. Appl.* **2015**, *128*, 23–28.
24. Awan, I.; Younas, M.; Naveed, W. Modelling QoS in IoT applications. In Proceedings of the 2014 International Conference on Network-Based Information Systems, NBiS 2014, Salerno, Italy, 10–12 September 2014; pp. 99–105, doi:10.1109/NBiS.2014.97.
25. Jin, J.; Gubbi, J.; Luo, T.; Palaniswami, M. Network architecture and QoS issues in the internet of things for a smart city. In Proceedings of the 2012 International Symposium on Communications and Information Technologies, ISCIT 2012, Gold Coast, QLD, Australia, 2–5 October 2012; pp. 956–961, doi:10.1109/ISCIT.2012.6381043.
26. Bouali, S.; Buscarino, A.; Fortuna, L.; Frasca, M.; Gambuzza, L.V. Emulating complex business cycles by using an electronic analogue. *Nonlinear Anal. Real World Appl.* **2012**, *13*, 2459–2465, doi:10.1016/j.nonrwa.2012.02.010.
27. Attwood, A.; Abuelmatti, O.; Fergus, P. M2M rendezvous redundancy for the internet of things. In Proceedings of the 2013 6th International Conference on Developments in ESystems Engineering, DeSE 2013, Abu Dhabi, United Arab Emirates, 16–18 December 2013; pp. 46–50, doi:10.1109/DeSE.2013.17.
28. Nukala, R.; Panduru, K.; Shields, A.; Riordan, D.; Doody, P.; Walsh, J. Internet of Things: A review from "Farm to Fork". In Proceedings of the 2016 27th Irish Signals and Systems Conference, ISSC 2016, Londonderry, UK, 21–22 June 2016; doi:10.1109/ISSC.2016.7528456.
29. Asif, M.; Khan, S.; Ahmad, R.; Sohail, M.; Singh, D. Quality of service of routing protocols in wireless sensor networks: A review. *IEEE Access* **2017**, *5*, 1846–1871, doi:10.1109/ACCESS.2017.2654356.

30. Dlodlo, N.; Kalezhi, J. The internet of things in agriculture for sustainable rural development. In Proceedings of the 2015 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC), Windhoek, Namibia, 17–20 May 2015; pp. 13–18, doi:10.1109/ETNCC.2015.7184801.
31. Nef, M.; Perlepes, L.; Stamoulis, G.I.G.; Karagiorgou, S.; Stamoulis, G.I.G.; Kikiras, P. Enabling QoS in the Internet of Things. In Proceedings of the CTRQ 2012: The Fifth International Conference on Communication Theory, Reliability, and Quality of Service, Chamonix/Mont Blanc, France, 29 April–4 May *2012*; pp. 33–38, doi:10.1128/JB.184.2.444-451.2002.
32. Li, L.; Li, S.; Zhao, S. QoS- Aware Scheduling of Services-Oriented Internet of Things. *IEEE Trans. Ind. Inform.* **2014**, *10*, 1497–1505, doi:10.1109/TII.2014.2306782.
33. Xu, W.; Ma, K.; Trappe, W.; Zhang, Y. Jamming sensor networks: Attack and defense strategies. *IEEE Netw.* **2013**, *20*, 41–47, doi:10.1109/MNET.2006.1637931.
34. Abdalzaher, M.S.; Seddik, K.; Elsabrouty, M.; Muta, O.; Furukawa, H.; Abdel-Rahman, A. Game theory meets wireless sensor networks security requirements and threats mitigation: A survey. *Sensors* **2016**, *16*, 22–27, doi:10.3390/s16071003.
35. Thomson, C. *Cooja Simulator Manual, (C), 2015–2016*; Edinburgh Napier University, Scotland 2016; doi:10.13140/RG.2.1.4274.8408.
36. Tarkowski, M.; Rzymowski, M.; Kulas, L.; Nyka, K. Improved jamming resistance using electronically steerable parasitic antenna radiator. In Proceedings of the 17th IEEE International Conference on Smart Technologies, EUROCON 2017, Ohrid, Macedonia, 6–8 July 2017; doi:10.1109/EUROCON.2017.8011161.
37. Gupta, H.; Vahid Dastjerdi, A.; Ghosh, S.K.; Buyya, R. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Softw. Pract. Exp.* **2017**, *47*, 1275–1296, doi:10.1002/spe.2509.