sensors

MDPI

Article

Congestion Control and Traffic Differentiation for Heterogeneous 6TiSCH Networks in IIoT

Hossam Farag *, Patrik Österberg and Mikael Gidlund

Department of Information Systems and Technology, Mid Sweden University, 851 70 Sundsvall, Sweden; patrik.osterberg@miun.se (P.Ö.); mikael.gidlund@miun.se (M.G.)
* Correspondence: hossam.farag@miun.se; Tel.: +46-010-142-8438

Received: 1 June 2020; Accepted: 19 June 2020; Published: 21 June 2020

Abstract: The Routing Protocol for Low power and lossy networks (RPL) has been introduced as the de-facto routing protocol for the Industrial Internet of Things (IIoT). In heavy load scenarios, particular parent nodes are likely prone to congestion, which in turn degrades the network performance, in terms of packet delivery and delay. Moreover, there is no explicit strategy in RPL to prioritize the transmission of different traffic types in heterogeneous 6TiSCH networks, each according to its criticality. In this paper, we address the aforementioned issues by introducing a congestion control and service differentiation strategies to support heterogeneous 6TiSCH networks in IIoT applications. First, we introduce a congestion control mechanism to achieve load balancing under heavy traffic scenarios. The congestion is detected through monitoring and sharing the status of the queue backlog among neighbor nodes. We define a new routing metric that considers the queue occupancy when selecting the new parent node in congestion situations. In addition, we design a multi-queue model to provide prioritized data transmission for critical data over the non-critical ones. Each traffic type is placed in a separate queue and scheduled for transmission based on the assigned queue priority, where critical data are always transmitted first. The performance of the proposed work is evaluated through extensive simulations and compared with existing work to demonstrate its effectiveness. The results show that our proposal achieves improved packet delivery and low queue losses under heavy load scenarios, as well as improved delay performance of critical traffic.

Keywords: Industrial IoT; 6TiSCH; RPL; trickle timer; priority; congestion; traffic differentiation

1. Introduction

The Industrial Internet of Things (IIoT) is the sub-category of the IoT that targets the industrial sector to improve productivity and efficiency [1]. Unlike consumer IoT, IIoT applications are characterized by stringent communication requirements in terms of reliability and delay [2]. The IPv6 over Time-Slotted Channel Hopping (6TiSCH) working group was established in 2013 with the aim to enable industrial-grade IPv6 networks to foster IIoT [3]. The 6TiSCH network is a key component to enable the adoption of IPv6 in industrial standards and the convergence of Operational Technology (OT) with Information Technology (IT) where industrial devices (e.g., sensors, actuators, robots, etc.) are enabled to connect to the cloud. The collected information from the industrial site is integrated via a control and management platform to improve the operational efficiency and productivity of the manufacturing process [4]. In this context, the 6TiSCH working group is developing solutions for missing components, such as schedule management, deterministic IPv6 flows, link management, and routing [5].

6TiSCH networks are Low-power and Lossy Networks (LLNs) that are constructed using low-cost and resource-constrained devices, namely LLN devices [6,7]. In 6TiSCH networks, communication routes are constructed and maintained through the Routing Protocol for LLN (RPL) [6]. RPL organizes the 6TiSCH network as a Destination-Oriented Acyclic Graph (DODAG) rooted at the sink node, namely the DODAG root. Each node is attached to a parent node that relays all its sensory information to the DODAG root. RPL employs the concept of RANKto define the relative position of a node with respect to the DODAG root. In RPL, the Objective Function (OF) describes the rules to compute the RANK and how it should be used to select the preferred parent [7]. Currently, there are two OFs defined by the RPL standard, OF zero (OF0) [8] and Minimum Rank with Hysteresis OF (MRHOF) [9]; however, there is no obligation to use a specific OF. Information about the RANK and OF is advertised through DODAG Information Object (DIO) messages, whose transmission interval is controlled by the Trickle timer algorithm [10]. Based on the information received by the DIO message and the adopted OF, the node selects a preferred parent from its candidate parent set. In heavy traffic load circumstances, parent nodes are likely to be congested due to the increased forwarding rate and the limited buffer size of LLN nodes. In the RPL standard, a parent node is selected either based on hop-counts or the link quality [6] without considering the congestion level causing an imbalanced network. Congestion has a profound impact on the network performance in terms of packet loss, delay, and energy consumption, and experimental measurements revealed that in high traffic scenarios, the main cause of packet loss is due to congestion [11]. The current RPL specifications do not specify how to detect and control congestion in 6TiSCH networks.

Moreover, in many industrial applications, such as monitoring and control scenarios, sensor nodes generate different traffic types with varying real-time requirements [12]. In certain scenarios, e.g., emergencies, critical traffic has higher importance than other types of traffic and must be delivered within a limited time to maintain stability, functionality, and to avoid dangerous situations. For data transmission scheduling, the current version of RPL considers the nodes to utilize either the First-In First-Out (FIFO) policy [13] or Last-In First-Out (LIFO) policy [14] in their output buffer. RPL has no explicit mechanism to efficiently handle the transmission of heterogeneous traffic based on the corresponding criticality and performance requirements. Accordingly, critical packets may be blocked and transmitted after non-critical ones, hence violating their timing limits. Therefore, traffic priority and network congestion are key research challenges in heterogeneous 6TiSCH networks in IIoT applications.

This paper presents a congestion control and service differentiation framework to support heterogeneous 6TiSCH networks in IIoT applications. Principally, our proposed approach investigates two key research questions: first, how to achieve fair load distribution in imbalanced 6TiSCH networks and achieve improved packet delivery performance in heavy traffic scenarios; second, how to guarantee prioritized and low delay transmissions of critical data over the non-critical in heterogeneous 6TiSCH networks. To this end, the first question is addressed through a congestion detection and control mechanism, while the second question is addressed through a multi-queue model, both combined in the proposed congestion control and service differentiation framework to achieve load balancing and improved packet delivery for all types of traffic and enhanced delay performance for critical data. To the best of our knowledge, this is the first work to address congestion and traffic differentiation issues in heterogeneous 6TiSCH networks. The key contributions of our proposed work can be summarized as follows

- We introduce a congestion control approach to achieve load balancing and improve network performance in terms of packet delivery under heavy load conditions. In the proposed approach, a new joint routing metric is defined to select parent nodes considering queue occupancy along with the hop distance and link quality metrics.
- To further support the functionality of the above strategy, we propose a Trickle timer reset strategy to detect overloaded nodes and to react to congestion in a timely fashion while maintaining minimum network overhead.
- Moreover, we design a multi-queue model where each node uses three different queues corresponding to three traffic categories, which is the typical case in most IIoT scenarios. Each queue is given a transmission priority where packets from higher priority queues are

transmitted first. In addition, we provide a stochastic mathematical model to formulate the average queue waiting time of the proposed multi-queue model.

- We evaluate the performance of the proposed work through extensive discrete-time simulations and conduct performance comparisons with existing work to demonstrate its effectiveness. The results show that our proposed framework achieves improved packet delivery and low queue losses under heavy load scenarios, as well as improved real-time performance of critical traffic.

The remainder of this paper is organized as follows. Section 2 discusses the problem statement and related work. The proposed congestion control method is introduced in Section 3. Section 4 presents the multi-queue model and its corresponding mathematical analysis. Performance evaluations are given in Section 5, followed by the conclusion and future work in Section 6.

2. Problem Statement and Related Work

In this section, we first highlight the congestion and traffic priority issues in RPL networks, then we discuss related works in this context.

2.1. Problem Statement

Although RPL has been designed to meet the requirements of LLNs, there are issues still challenging to satisfy the stringent requirements of heterogeneous 6TiSCH networks in IIoT applications. This paper mainly tackles two issues.

The first issue is the congestion control under heavy traffic load and imbalanced DODAG construction. RPL is mainly designed to handle traffic in LLNs under light traffic conditions. However, in high traffic conditions, parent nodes are prone to congestion problems, especially those close to the DODAG root. In RPL, the considered OF allows each node to select its preferred parent based on the hop-count and link quality, i.e., Expected Transmission Count (ETX), regardless of the queue occupancy of neighboring nodes. Hence, the OF does not reflect the congestion of parent nodes, which in turn degrades network performance in terms of packet loss and delay. The problem even exists under light traffic condition where there is unfair load distribution among parent nodes.

To better understand the problem, consider the routing topology shown in Figure 1a. Node C is overloaded with children compared to nodes A, B, and D, which have the same RANK. This in turn incurs a significant traffic load at node E, the parent node of node C, which is responsible for forwarding the traffic of more than 50% of the network. In high traffic conditions, the children of C send packets with high rates, which leads to buffer overflow at node C, while nodes A, B, and D maintain a stable buffer status. The same, and even worse, occurs at node E. Typically, LLN devices are resource-constrained and have small queue sizes. These small queues start to overflow before the congestion becomes heavy enough to be detected through the ETX and Trickle timer. The children of C are not aware of the congestion problem, and hence, they continue to transmit packets to C as they measure low ETX from their perspective. Moreover, the Trickle timer is not aware of the congestion situation; thus, the node cannot change its congested parent in a timely manner. Therefore, the conventional Trickle timer and using ETX for parent selection is not reasonable for load balancing in RPL-based networks. In Figure 2, we investigate packet losses in an RPL network with OF0 under different traffic load conditions. From this figure, we can clearly note that queue losses constitute the dominant part of packet losses compared to channel losses (almost 16%) even in high traffic load conditions. This in turn will be misleading for OF0 to change the parent when the node is congested, thus the need for an efficient parent selection and congestion control mechanism.

The second issue is the traffic priority in heterogeneous 6TiSCH networks. RPL defines no mechanism to prioritize the transmission of heterogeneous traffic types in 6TiSCH networks according to their timing requirements. All packets are placed in a single queue and scheduled either in an LIFO or FIFO fashion. High priority data are mainly characterized by tight timing constraints, and if they arrive too late, they are of limited use and could lead to system failure, production loss, or even dangerous situations [12]. Both queue scheduling policies adopted by the RPL may cause higher priority transmissions to violate its timing limits when blocked by the transmission of lower priority ones, which typically have relaxed timing requirements. Therefore, the queue scheduling of the current RPL version is an inefficient solution in IIoT applications with heterogeneous traffic. To further illustrate the problem, consider the 6TiSCH network scenario depicted by Figure 1a. As a result of a particular emergency event, an emergency packet arrives at the output queue of node F at time $t = i$, as shown in Figure 1b. Due to its criticality, this packet should be transmitted first with the highest priority. At $t = i + 1$, a regular packet joins the output queue of the same node that is either coming from one of its children or generated by node F itself. Considering the LIFO policy for this case, at $t = i + 2$, the emergency packet transmission is blocked due to the transmission of the recently arrived regular packet, which may cause the emergency packet to miss its deadline. A similar situation would happen if we consider the FIFO policy. Therefore, for such applications, a proper packet scheduling method is needed.

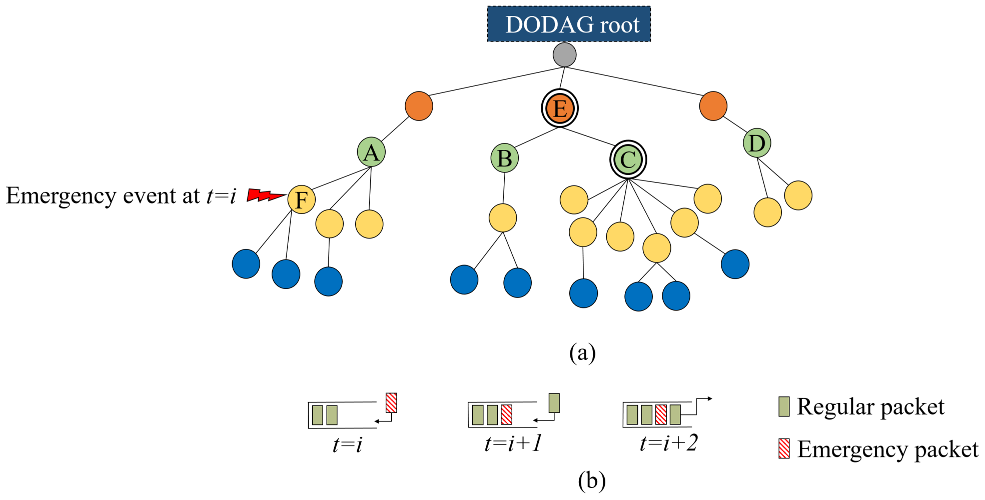


Figure 1. 6TiSCH network scenario: (**a**) routing topology; (**b**) LIFO queue model of node *F*.

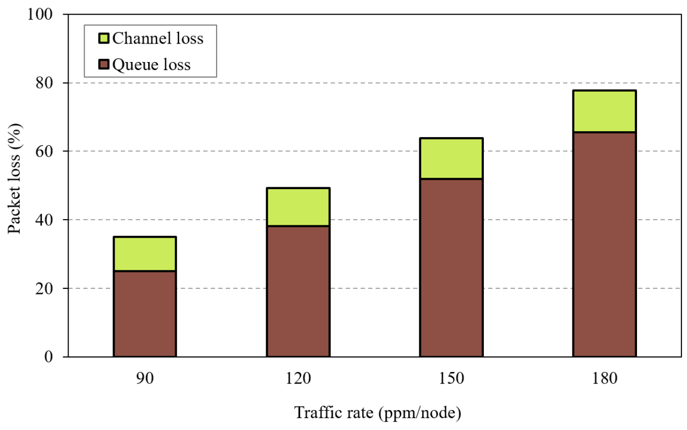


Figure 2. Packet loss for different traffic rates.

2.2. Related Work

In recent years, several works were introduced to improve the performance of RPL-based networks in terms of different metrics [14–16]. However, most of these efforts overlook reliability and real-time aspects, which are crucial for IIoT applications especially when involving transmission of heterogeneous data. Routing and data transmission scheduling are the key components that directly affect data transport capabilities in terms of reliability and real-time delivery. In the context of data transmission scheduling, a number of research efforts were proposed to support the real-time transmission of critical data in industrial applications [12,17–19]. These works are mainly based on improved channel access mechanisms that allow critical data to gain higher priority to access the channel over the non-critical ones. However, those works are only applicable for single-hop networks. However, it has been shown that the multi-queuing strategy plays a major role in affecting the Quality-of-Service (QoS) requirements of wireless sensor networks with multiple traffic types [20]. The work in [21] introduced EARS, an emergency packet scheduling scheme for IoT in smart cities. In EARS, each incoming packet is placed in the corresponding queue based on its priority and deadline information where emergency packets are always processed and transmitted first. A dynamic multilevel priority packet scheduling scheme was proposed in [22], where each node had three levels of priority queues. In this approach, real-time/emergency packets are placed in the highest priority queue and can preempt other queues, while non-real time data are placed into the other two queues and processed based on the shortest job first scheduler. Another multilevel queuing approach was proposed in [23] where each node utilizes a number of queues based on its location. The node decides the packet priority based on its hop count, and accordingly, the packet is placed in the relevant queue. In [24], the authors proposed a cloud-assisted priority-based scheme. The prioritized data packets received by each cluster head are sent to one of two queues: the high priority queue or the low priority queue where the preemptive M/G/1queuing model is employed. All the aforementioned multi-queuing approaches have been shown to perform well in light traffic conditions; however, the negative impact of congestion in heavy traffic scenarios in 6TiSCH networks is not considered, which has a direct effect on the reliability real-time communications of high priority traffic.

Several works have been introduced to control the congestion in RPL-based networks. The MLEqprotocol [25] was proposed to handle the congestion problem using multiple gateways in IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN). Congested gateways share their load in a distributed fashion to achieve a global load fairness; however, the approach is not applicable in RPL networks with a single gateway. The works in [26–28] proposed to formulate the congestion issue as an optimization problem based on a strictly concave and continuously differentiable function, e.g., congestion cost function or a utility function, where the objective is to select the optimal sending rate to minimize the congestion level. However, this approach is not applicable in many IIoT applications where setting a predefined sending rate is crucial to maintain the network functionality and stability. The authors in [29] proposed M-RPL, a Multi-path extension of RPL to alleviate the congestion problem by providing temporary multiple paths for the congested nodes. Although M-RPL manages to improve the network performance in terms of energy consumption and throughput, it suffers from increased delay when constructing the multi-path routes. Moreover, the conventional Trickle timer utilized by the aforementioned approaches could fail to detect and react to congestion in a timely manner. The authors in [30] proposed CoAR, a Congestion-Aware Routing protocol, where the Trickle timer is reset to its minimum value when a congestion is detected. Although CoAR improves the network performance in congestion situations, the utilized reset strategy increases the network overhead and the energy consumption in turn. The authors in [31] proposed a heuristic algorithm that calculates the redundancy constant of the Trickle timer as a function of the number of neighbor nodes in its vicinity. However, the work overlooked the impact of the proposed algorithm on the QoS metrics of the constructed routes. The Trickle-Dalgorithm was proposed in [32], in which the redundancy constant is adapted using Jain's index to achieve fairness among nodes while keeping low overhead. According to the obtained results, Trickle-D achieves improved performance in terms of fairness and energy consumption, while other vital metrics such as packet delivery ratio and delay are not considered. Moreover, none of the above works considered the prioritized transmission of critical traffic in heterogeneous 6TiSCH networks.

3. The Proposed Congestion Control Mechanism

In this section, we describe the congestion control framework used to achieve load balancing in 6TiSCH networks. We first present a congestion detection and control method, then we describe a mechanism to distribute congestion information between neighbor nodes. Finally, we introduce the proposed Trickle timer reset strategy.

3.1. Detecting and Controlling Congestion

Initially, the DODAG is constructed through exchanging the DIO messages between neighbor nodes. First, a node n_i generates its parent candidate set $Parent(n_i)$ as a subset of its neighbor candidate set $N(n_i)$ as follows:

$$Parent(n_i) = \left\{ n_j \in N(n_i) | H(n_j) < H(n_i), ETX(n_i, n_j) < \gamma \right\}, \tag{1}$$

where $H(n_j)$ denotes the hop-count between n_j and the DODAG root, $ETX(n_i, n_j)$ is the estimated ETX value between n_i and n_j, and γ is a threshold to eliminate neighbors with bad link quality. $ETX(n_i, n_j)$ is obtained as [33]:

$$ETX(n_i, n_j) = \frac{\#\,\text{of total transmissions from}\, n_i \text{ to } n_j}{\#\,\text{of successful transmissions from}\, n_i \text{ to } n_j}. \tag{2}$$

Each node n_i selects a preferred parent node P_i from its $Parent(n_i)$ for data forwarding. We consider that the initial P_i is selected as the one that has the minimum hop-distance towards the DODAG root [8].

A node n_i selects a new parent P_i^* when changes occur to its $Parent(n_i)$. In our proposed method, the new parent selection process is triggered if one of the following two criteria are satisfied: the joint Hop-distance and Link quality (HL)-criterion, and the Load-Balancing (LB)-criterion. Based on these criteria, we also introduce two distinct parent selection mechanisms.

3.1.1. The HL-Criterion

The first selection method corresponds to satisfying only the HL-criterion. This case represents light traffic scenarios and mainly aims to select P_i^* based on link quality and hop-count. The HL-criterion is defined as:

$$R^{HL}(P_i) - R^{HL}(P_i^*) > \theta, \tag{3}$$

where $R^{HL}(P_i)$ is the routing metric with respect to P_i and θ is the hysteresis value used to avoid excessive parent switching due to small changes in the routing metric. The HL-criterion is based on the routing metric $R^{HL}(p_i)$ that reflects the link quality and the hop distance for each parent candidate $p_i \in Parent(n_i)$. $R^{HL}(p_i)$ is given as:

$$R^{HL}(p_i) = \underbrace{H(p_i) + 1}_{RANK(p_i)} + ETX(n_i, p_i). \tag{4}$$

When the HL-criterion in Equation (3) is satisfied, a new parent P_i^* is selected as:

$$P_i^* = \min_{p_i \in Parent(n_i)} \left\{ R^{HL}(p_i) \right\}. \tag{5}$$

Hence, in light traffic conditions, a node selects its new parent mainly based on the link quality and the hop-distance.

3.1.2. The LB-Criterion

As mentioned earlier, the ETX metric is insufficient for the detection of congestion in heavy traffic conditions. The small queues of the LLN nodes start to overflow before the congestion is heavy enough to degrade the ETX and be detected through the HL-criterion. In this case, n_i keeps transmitting to its P_i even if it suffers from consecutive queue losses.

We introduce the LB-criterion to detect the congestion in parent nodes and change to P_i^* based on the queue occupancy information to achieve load balancing. The formulation of the LB-criterion is mainly based on the backlog factor $BF(n_i)$ that represents the queue occupancy of n_i. $BF(n_i)$ is defined as the ratio of the number of backlogged packets in the output queue $Q(n_i)$ to the total queue size $L(n_i)$. How to set the LB-criterion and $BF(n_i)$ properly is illustrated as follows.

When P_i is congested, $BF(n_i)$ may be much smaller than $BF(P_i)$ even if P_i suffers from queue losses; hence, $BF(n_i)$ cannot properly reflect congestion in this case. Furthermore, when $BF(n_i)$ is high, changing the parent of n_i would not help in load balancing, but instead, the children of n_i should migrate to another parent to reduce the load on n_i. $BF(P_i)$ is also not a proper indicator of congestion, as when the network is balanced, each node will have low $BF(P_i)$; hence, the LB-criterion would not be satisfied, and parent selection would only be triggered based on the HL-criterion, causing an imbalanced network again. Therefore, we define the LB-criterion as:

$$\max\left\{BF_{max}, BF_{max}^{[m+1]}(n_i)\right\} > \delta, \tag{6}$$

where BF_{max} is the maximum backlog factor recognized for all parent candidates of n_i within the last m consecutive slotframes, $BF_{max}^{[m+1]}(n_i)$ is the maximum backlog factor of all parents of n_i in the current slotframe, and δ is the congestion threshold. To further illustrate, the maximum backlog factor of parent nodes recorded by node n_i within the j^{th} slotframe is given as:

$$BF_{max}^{[j]}(n_i) = \max_{p_i \in Parent(n_i)} \left\{BF(p_i)\right\}.$$

Then, n_i maintains these values and calculates the maximum for the recent m slotframes as follows:

$$BF_{max} = \max_{j \in \{1,2,\ldots,m\}} \left\{BF_{max}^{[j]}(n_i)\right\}.$$

Finally, n_i uses the sliding window to update BF_{max} for every slotframe, select the maximum of these values, and compare it with the congestion threshold δ as given in (6).

The value of the number of consecutive slotframes m in (6) is a configurable parameter and is empirically selected considering the following conditions regarding the selection of its lower and upper limits. First, m is selected such that $(m \times T) > I_{min}$, where T is the duration of a slotframe and I_{min} is the minimum interval of the Trickle timer. This is because the value of $BF_{max}^{[j]}(n_i)$ of a node n_i is determined using the backlog information of its parents $BF(p_i)$ according to (6), and such information is propagated through the DIO messages whose interval is determined through the Trickle timer, i.e., collecting the information at least every I_{min}. Moreover, the node needs to maintain a reasonable record of past congestion events in order to be more aware of the congestion situation of its parents and avoid hasty parent changes. Second, LLN nodes are resource-constrained devices that typically have limited storage capacity, and increasing m means storing more values of $BF(p_i)$, which might be a limitation to the node if it increases beyond a certain value. Therefore, the upper bound of m is mainly based on the available resources. Furthermore, it is pointless to increase m to store too old history of $BF(p_i)$.

The threshold value $0 < \delta < 1$ determines when to perform parent change as a result of congestion. Empirically, we consider $\delta = 0.5$, which means that a new parent P_i^* should be selected when the congestion level is above 50%. A higher value of δ could cause a delayed detection of congestion. However, a lower value of δ could trigger unnecessary parent change actions when the network can support the current traffic load, which incurs increased overhead.

Next, we introduce a new routing metric $R^{LB}(p_i)$ to consider load balancing when selecting P_i^*:

$$R^{LB}(p_i) = \underbrace{H(p_i) + 1}_{RANK(p_i)} + ETX(n_i, p_i) + \lambda BF(p_i), \tag{7}$$

where λ is a weighting coefficient that controls the effect of $BF(p_i)$ on the parent selection process. Since we have $0 \leq BF(p_i) \leq 1$, we should have $\lambda > 1$ for $BF(p_i)$ to have a notable effect compared to $H(p_i)$ and ETX factors. The impact of λ is discussed later in Section 5. Then, when a congestion is detected, i.e., Equation (6) is satisfied, a new parent P_i^* is selected as:

$$P_i^* = \min_{p_i \in Parent(n_i)} \left\{ R^{LB}(p_i) \right\}. \tag{8}$$

When congestion occurs at P_i, it is better not to select n_i as a parent by its neighbors since all its traffic is eventually forwarded to P_i, which is already congested. To address this case, n_i updates its $BF(n_i)$ after each parent selection as:

$$BF(n_i) = \max \left\{ BF(P_i) - \Delta, \frac{Q(n_i)}{L(n_i)} \right\}, \tag{9}$$

where $0 < \Delta < 1$ is a small factor. This way, children nodes can be aware of those congested ancestors located up to $\left(\lceil \frac{1}{\Delta} \rceil - 1 \right)$ hops and avoid selecting parents directly connected to them.

When utilizing only the LB-criterion to detect the congestion and select a new parent, a number of nodes may simultaneously change to the same parent with the minimum routing metric according to Equation (8). This in turn causes a congestion problem for that new parent. Accordingly, these nodes detect the congestion and change their parent once more, with the minimum backlog factor resulting in congestion again. This may lead to an indefinite cycle of parent changes without achieving a balanced network, a phenomenon that is known as the thundering herd problem [34]. This problem can be illustrated by the scenario shown in Figure 3. To the left, node B is congested since it is attached to many children nodes. According to Equations (7) and (8), the children nodes handle the congestion event by changing to a new non-congested parent. The problem is that all the children nodes may simultaneously change to the same new parent, i.e., node A as shown in the right side of the figure, which again results in a congestion in that node. In that case, those nodes may continue to change their parent indefinitely without achieving load balancing.

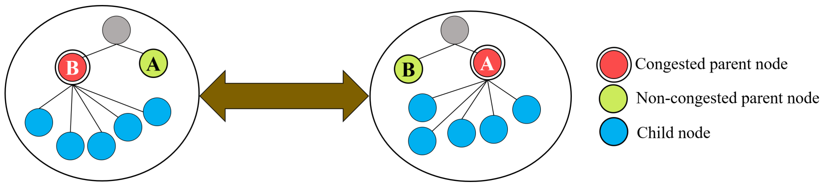


Figure 3. Example of the thundering herd problem.

To evade this problem, we introduce a probability-based parent switching mechanism. When the LB-criterion in Equation (6) is satisfied, a node chooses to change to a new parent according to Equation (8) with the following probability:

$$P_{switch} = \max \left\{ \Gamma \left(BF\left(P_i\right) - BF\left(P_i^*\right) \right), 0 \right\}, \tag{10}$$

where $0 < \Gamma < 1$ is a small coefficient that represents the node combativeness to change its parent to avoid congestion. The effect of Γ on the network performance is evaluated in Section 5.

When both the HL-criterion and the LB-criterion are satisfied, the selection mechanism in Equation (8) is applied, because in this case, balancing the network load is more important, as mentioned earlier.

3.2. Exchanging the Queue Backlog Information

Neighbor nodes need to share their queue backlog information, i.e., $BF(n_i)$, in order to be aware of the congestion status. To do so in the method we propose, the value of $BF(n_i)$ is implicitly embedded into the RANK field in the DIO message in the RPL standard [6]. Accordingly, we change the definition of the RANK in the DIO message to:

$$RANK_{new}(n_i) = \eta\,(H(n_i)+1) + (\eta-1)BF(n_i), \tag{11}$$

where η is a decoding factor to decode the value of $BF(n_i)$ (single value) from $RANK_{new}$ (two values). η can be any positive integer value that keeps $RANK_{new}$ within its 16 bit boundary [6]. When a neighbor node receives the DIO message of n_i, the values of $BF(n_i)$ and $H(n_i)$ are decoded separately as follows:

$$\begin{aligned} BF(n_i) &= \frac{mod\,(RANK_{new}(n_i),\eta)}{\eta-1}, \\ H(n_i) &= \left\lfloor \frac{RANK_{new}(n_i)}{\eta} \right\rfloor - 1, \end{aligned} \tag{12}$$

where *mod* () is the modulo operation. This way, the queue backlog information is distributed among neighbor nodes without the need to change the DIO message format, which ensures that the proposed scheme is compliant with the standard RPL.

3.3. Modified Trickle Timer Algorithm

The backlog information should be distributed through the DIO message in a timely manner in order to detect and react quickly to congestion. As mentioned in Section 1, the standard RPL uses the Trickle timer to control the transmission interval of DIO messages. As long as the network is consistent, the DIO message interval is doubled up to a certain maximum value. The timer is reset to a minimum value when inconsistency is detected [10]. However, when the network is consistent, the long DIO interval may cause the nodes to have inaccurate and outdated congestion information, hence fail to achieve load balancing.

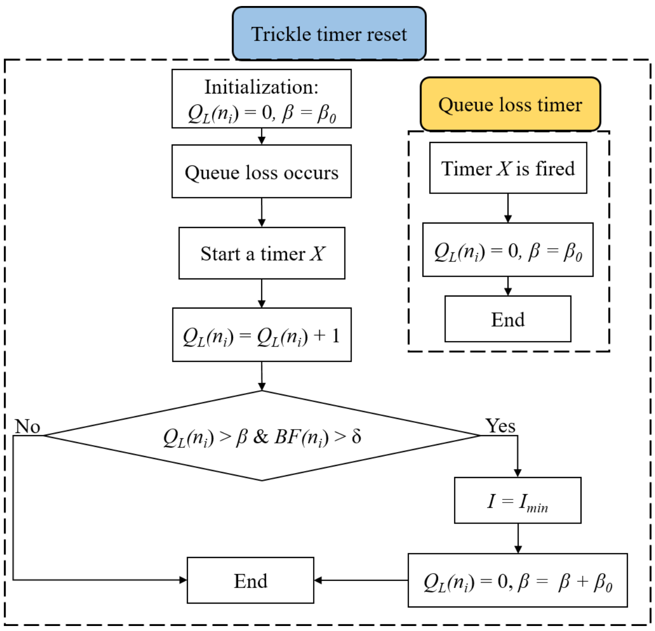

Trickle timer reset
Initialization: $Q_L(n_i) = 0, \beta = \beta_0$
Queue loss occurs
Start a timer X
$Q_L(n_i) = Q_L(n_i) + 1$
$Q_L(n_i) > \beta$ & $BF(n_i) > \delta$
No
Yes
$I = I_{min}$
$Q_L(n_i) = 0, \beta = \beta + \beta_0$
End
Queue loss timer
Timer X is fired
$Q_L(n_i) = 0, \beta = \beta_0$
End

Figure 4. Modified Trickle timer algorithm.

To address this issue, we propose a modification to the Trickle timer in order to distribute the backlog information in a timely manner while keeping the control message overhead to a minimum. The flowchart of the modified Trickle timer algorithm is shown in Figure 4 and described as follows.

The basic idea is to reset the Trickle timer interval when the node suffers a certain number of consecutive queue losses $Q_L(n_i)$. The intention behind using $Q_L(n_i)$ is that LLN nodes have small queues that may fill up temporarily even when there is no congestion in the network. This temporary situation may trigger a false congestion that leads to unnecessary overhead, which can be avoided if we reset the Trickle timer after detecting a particular number of consecutive queue losses. The Trickle timer is reset to its minimum interval I_{min} [10] when $BF(n_i)$ exceeds δ and $Q_L(n_i)$ exceeds a certain limit β. Once a queue loss occurs, a timer X is triggered. It is used as a timeout period, that is if no queue losses are detected within this period, the parameters $Q_L(n_i)$ and β are reinitialized. When the Trickle timer is reset to I_{min}, the value of β is increased by a minimum value β_0 to decrease strictly the number of times the Trickle timer is reset, i.e., minimize the overhead. Therefore, the proposed reset strategy allows the nodes to acquire the queue backlog information through the DIO messages in a timely fashion to act hastily to congestion events, while keeping the DIO messages overhead to a minimum.

The proposed modification in Trickle timer adds insignificant computational complexity. In terms of memory resources, each node needs to store $Q_L(n_i)$, $BF(n_i)$, and β, whose values are not too demanding for storage. In terms of the number of elementary operations, a single sum operation is added, which is executed upon Trickle timer reset, a single increment operation that is executed upon packet loss, and a single subtract operation that is executed when the timer X is fired. That is, the computational complexity of the modified reset strategy is $O(1)$.

4. Multi-Queue Model and Priority-Based Transmission

In this section, we introduce a priority-based multi-queue transmission model in order to support heterogeneous traffic in 6TiSCH networks. Then, we derive a mathematical model to formulate the average waiting time in each queue.

4.1. The Multi-Queue Transmission Model

As discussed in Section 2, the conventional single queue model in 6TiSCH networks cannot guarantee the real-time requirements of high priority data in IIoT applications. To deal with this issue, we design a multi-queue model to support traffic differentiation in heterogeneous 6TiSCH networks in IIoT applications.

Each node exploits a different output queue for each traffic type. We consider a 6TiSCH network that supports up to three traffic types:

- T_1: represents the safety-critical traffic that has the highest priority, e.g., fire alarms and emergency shutdown.
- T_2: denotes the acyclic control traffic, which is often time critical. T_2 has lower priority than T_1, but higher priority than T_3.
- T_3: represents periodic monitoring traffic that is less critical and generated at predictable time instants, e.g., periodic temperature measurements. T_3 has the lowest priority with relaxed timing requirements.

In our proposed multi-queue model, each node maintains three equally-sized queues, Q_1, Q_2, and Q_3 for T_1, T_2, and T_3, respectively. The packets in Q_1 are given the highest priority and are always transmitted first, followed by the packets in Q_2, and lastly, Q_3, which is given the lowest priority.

In order to elaborate our proposed multi-queue model, we consider an industrial real-world scenario of heterogeneous traffic, which is the process monitoring of plastic extrusion [35]. Plastic extrusion is a high volume manufacturing process in which raw plastic material is melted and formed into a continuous profile to form product items such as pipe/tubing, weather stripping, window frames, adhesive tape, and wire insulation. Firstly, it is critically important to measure pressure in the extruder to prevent serious accidents that can happen when excessively high pressures are generated [36]. Exceeding the safety pressure threshold may ultimately cause an explosion, the barrel may crack, or the die may be blown from the extruder. Melt temperature is also one of the most important variables that has to be maintained very carefully to produce a good quality product. It is important to ensure that the melt is not degraded or overheated during extrusion. A tight temperature profile along the barrel is very important for obtaining a constant quality of plastic material [36]. The whole process could be controlled remotely where all the collected information is transmitted to the Internet through the DODAG root. To map the different traffic within the plastic extrusion scenario to our defined multi-queue model, we have: T_1 refers to the safety alarms that are generated when the extruder pressure exceeds the predefined threshold to either inform the operator to shut-down the extruder or initiate an automatic shut-down command. T_2 refers to the control traffic generated when a notable deviation is detected in the temperature profile readings, and this in turn initiates a temperature control mechanism to avoid excessive economical loss. Lastly, T_3 represents the periodic temperature measurements from the thermocouple temperature sensors.

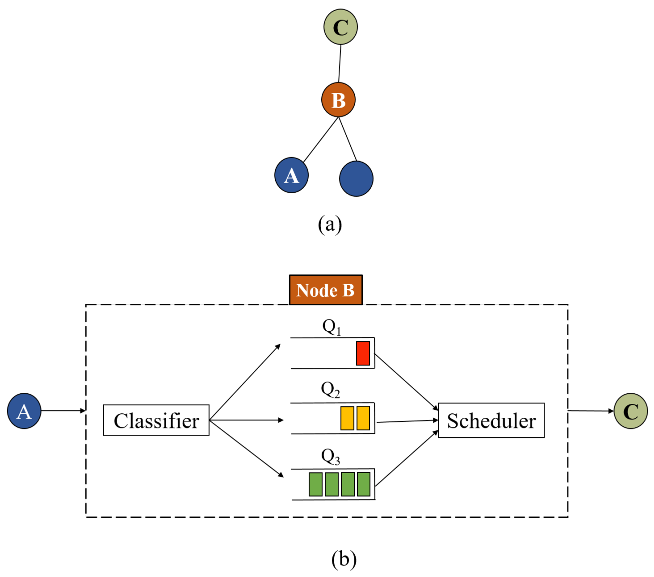


Figure 5. The proposed multi-queue model: (**a**) sub-tree of the Destination-Oriented Acyclic Graph (DODAG); (**b**) multi-queue model of node *B*.

The proposed queuing model is shown in Figure 5, and its working principle is described as follows. A classifier is responsible of sorting the incoming heterogeneous traffic either generated from the node itself or incoming from its children. Based on type and priority, i.e., T_1, T_2, or T_3, the packet is placed in the corresponding queue. The scheduler selects the packet to be transmitted first according to the priority of each queue. If there are T_1 packets in Q_1, the scheduler selects the packet to be transmitted first according to the Earliest-Deadline-First (EDF) approach [37], i.e., the T_1 packet with the minimum absolute deadline is selected. The absolute deadline D_i of an arbitrary packet $\mathrm{Pk_i}$ equals to its arrival time t_i plus the relative deadline d_i, i.e., $D_i = t_i + d_i$. In this context, the relative deadline of a packet is assigned according to the considered application and the information carried in the corresponding packet, e.g., fire alarm, excessive pressure in a pipe, leakage of gas, etc. If Q_1 is empty, the scheduler selects a packet from Q_2 to be transmitted first according to the EDF approach. If both Q_1 and Q_2 are empty, the packets from Q_3 are transmitted according to the FIFO policy. Therefore, the packet delay and queuing time are mainly influenced by the number of higher priority packets in the system, as will be illustrated in the next section. The packet category can be embedded in the packet header as a 2 bit field, i.e., 00, 01, and 10 for T_1, T_2, and T_3, respectively, which can be extracted and decoded by the classifier to place the packet in its corresponding queue.

This way, the proposed multi-queue model always ensures a prioritized transmission for the critical data over the non-critical compared to the conventional single-queue model. Another important advantage of the proposed multi-queue scheme is that higher priority data avoid the congestion problem. Traffic types T_1 and T_2 occur occasionally; hence, Q_1 and Q_2 are likely less prone to congestion problems at parent nodes, i.e., queue overflow and packets loss. Further, this guarantees reliable and real-time communications of critical data even under the heavy traffic load of T_3.

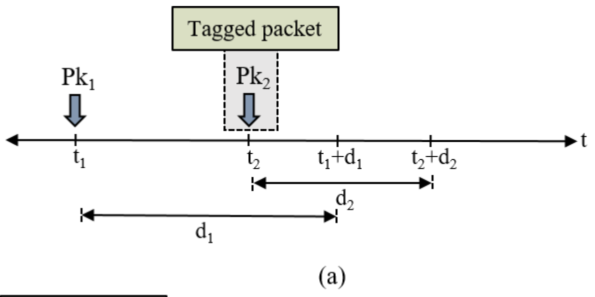
4.2. Mathematical Analysis

In the following, we present a queuing analysis of the proposed multi-queue model where we mathematically formulate the average queue waiting time $\overline{W_{Q_i}}$. The formulated $\overline{W_{Q_i}}$ corresponds to an arbitrary packet in Q_i, which will be later referred as the tagged packet. The average queue waiting time of a packet is defined as the time elapsed from the instant the packet is received by the node until the instant it leaves the queue. Each queue in the proposed multi-queue model is represented by a finite-capacity queuing system with a finite storage of K [31]. The arrival rate to each queue is modeled based on the generation nature of the corresponding traffic. Since T_1 and T_2 are acyclic in nature, we model the packet arrivals to Q_1 and Q_2 as a Poisson process with parameters α_1 and α_2, respectively. T_3 traffic is cyclic; thus, packets arrive at Q_3 periodically with a rate α_3. In the TSCH schedule, the service time for all packets in each queue is deterministic and equals T_c [38], where T_c is the horizontal length of the TSCH schedule, i.e., the number of time slots per channel. Therefore, Q_1 and Q_2 represent an M/D/1/Kqueuing system, while Q_3 represents a D/D/1/Kqueuing system [39].

According to our multi-priority model, packets from Q_1 are always transmitted first following the EDF scheduling policy, i.e., the packet with the shortest deadline will be transmitted first. We consider the general case that the packets in Q_1 and Q_2 may join the queue with different deadlines. For instance, considering two packets of the T_1 type in oil and gas industries, a packet that corresponds to a fire alarm may have a deadline that is different from that of a packet generated to alert about excessive pressure in a pipe. However, our proposed analysis can be simplified to a simple case of all packets within the same traffic category having the same deadline. The basic idea in the following mathematical formulations is to estimate the average number of higher priority packets with respect to an arbitrary tagged packet that would be transmitted ahead of it.

4.2.1. Average Queue Waiting Time in Q_1

Based on the aforementioned considerations, since the transmission priority of a T_1 packet within Q_1 is determined based on the absolute deadline, out of the packets already in Q_1, there are N_1^B packets whose absolute deadlines are earlier than that of the tagged packet and scheduled for transmission before it.



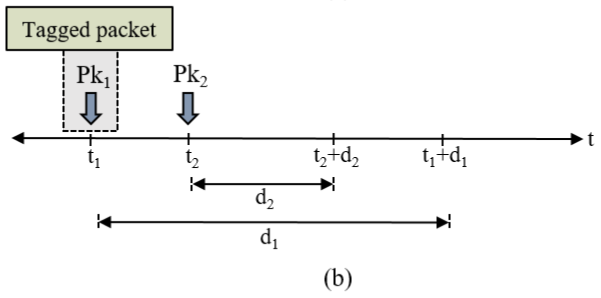


Figure 6. Packet arrivals in the proposed multi-queue model: (**a**) scenario of N_1^B; (**b**) scenario of N_1^A.

This scenario is illustrated by Figure 6a, where we have a packet Pk_1 with a relative deadline d_1 and our tagged packet Pk_2 with a relative deadline d_2. According to the EDF policy, the transmission priority of each packet within Q_1 is assigned upon its arrival based on its absolute deadline $(t + d_i)$. Although $d_2 < d_1$, Pk_1 has a higher transmission priority than Pk_2, since the former has an earlier absolute deadline. This applies to the packets arriving at least $(d_1 - d_2)$ before the arrival of Pk_2 and has been waiting for at least $(d_1 - d_2)$ given that $(d_1 - d_2) > 0$. Therefore, we have:

$$\overline{N_1^B} = \max\left\{0, \overline{\alpha_1}\left(\overline{W_{Q_1}} - D_{Q_1}\right)\right\}, \tag{13}$$

where $\overline{\alpha_1}$ is the effective arrival rate (we will derive the formula of $\overline{\alpha_1}$ later in this section), and $D_{Q_1} = d_1 - d_2$, which can be deterministic for constant values of d_1 and d_2.

In addition, there are a number of N_1^A packets that arrive after the tagged packet with an earlier absolute deadline, hence are transmitted first. This case is depicted in Figure 6b, where packet Pk_2 arrives after the tagged packet Pk_1, with an earlier absolute deadline. This applies to all packets that arrive after the tagged packet no later than D_{Q_1}. However, the tagged packet may stay in Q_1 for a period less than D_{Q_1}, given that $\overline{W_{Q_1}} < D_{Q_1}$. Thus, $\overline{N_1^A}$ is given as:

$$\overline{N_1^A} = \overline{\alpha_1}\min\left\{\overline{W_{Q_1}}, D_{Q_1}\right\}. \tag{14}$$

When those $(N_1^B + N_1^A)$ packets are transmitted and the tagged packet becomes the one with the highest priority in Q_1, it will wait for additional time until the beginning of its assigned slot, which is on average $\frac{1}{2}T_c$ [38]. Therefore, $\overline{W_{Q_1}}$ can be calculated as:

$$\begin{aligned}\overline{W_{Q_1}} &= \left(T_c + \frac{1}{2}T_c\right)\left(\overline{N_1^B} + \overline{N_1^A}\right) + \frac{1}{2}T_c \\ &= \frac{T_c}{2}\left(3\left(\overline{N_1^B} + \overline{N_1^A}\right) + 1\right).\end{aligned} \tag{15}$$

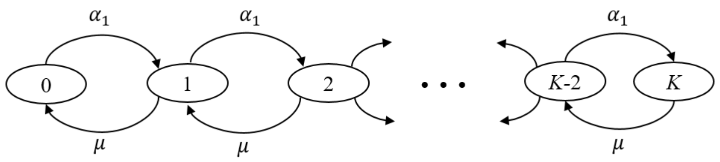


Figure 7. The state-transition diagram for the finite Markov chain of Q_1.

The next step is to estimate the value of $\overline{\alpha_1}$. Since Q_1 can hold at most K packets, T_1's packets will continue to be generated according to a Poisson process with parameter α_1; however, only the packets that find Q_1 with strictly less than K packets will be allowed to join. This is illustrated by Figure 7, which depicts the state-transition diagram for the finite Markov chain of Q_1, where $\mu = (1/T_c)$ denotes the service rate. The system can be modeled as a birth-death process [39] where the Poisson input is turned off as soon as Q_1 is filled up. Therefore, we have:

$$\alpha_{1k} = \begin{cases}\alpha_1 & k < K \\ 0 & k \geq K,\end{cases} \tag{16}$$

where α_{1k} is the birth rate at Q_1 when it includes k packets. It is worth mentioning that the system in Figure 7 and the following derivations are also valid for Q_2 and Q_3. Using Equation (16), the effective arrival rate $\overline{\alpha_1}$ can be given as:

$$\overline{\alpha_1} = \sum_{k=0}^{K}\alpha_{1k}\,p_k = \sum_{k=0}^{K-1}\alpha_1 p_k, \tag{17}$$

where p_k is the steady-state probability to find k packets in Q_1. Solving the equilibrium equations for the queuing system in Figure 7, we obtain:

$$p_k = \begin{cases} p_0\left(\frac{\alpha_1}{\mu}\right)^k & 0 \le k \le K \\ 0 & k > K. \end{cases} \tag{18}$$

Using Equation (18) along with the conservation relation $\sum_{k=0}^{\infty} p_k = 1$, we solve for p_0:

$$\begin{aligned} p_0 &= \frac{1}{1+\sum_{k=1}^{K}\left(\frac{\alpha_1}{\mu}\right)^k} = \left[1+\frac{\left(\frac{\alpha_1}{\mu}\right)\left(1-\left(\frac{\alpha_1}{\mu}\right)^K\right)}{1-\frac{\alpha_1}{\mu}}\right]^{-1} \\ &= \frac{1-\frac{\alpha_1}{\mu}}{1-\left(\frac{\alpha_1}{\mu}\right)^{(K+1)}}. \end{aligned} \tag{19}$$

Then, p_k in Equation (18) can be rewritten as:

$$p_k = \begin{cases} \frac{\left(\frac{\alpha_1}{\mu}\right)^k\left(1-\frac{\alpha_1}{\mu}\right)}{1-\left(\frac{\alpha_1}{\mu}\right)^{(K+1)}} & 0 \le k \le K \\ 0 & k > K. \end{cases} \tag{20}$$

Applying Equation (20) in Equation (17), we get:

$$\begin{aligned} \overline{\alpha_1} &= \alpha_1 \sum_{k=0}^{K} \frac{\left(\frac{\alpha_1}{\mu}\right)^k\left(1-\frac{\alpha_1}{\mu}\right)}{1-\left(\frac{\alpha_1}{\mu}\right)^{(K+1)}} \\ &= \alpha_1 \frac{1-\left(\frac{\alpha_1}{\mu}\right)^K}{1-\left(\frac{\alpha_1}{\mu}\right)^{(K+1)}} \\ &= \alpha_1 \frac{1-(\alpha_1 T_c)^K}{1-(\alpha_1 T_c)^{(K+1)}}. \end{aligned} \tag{21}$$

Based on Equations (13)–(15), and (21), $\overline{W_{Q_1}}$ can be given as:

$$\overline{W_{Q_1}} = \frac{T_c}{2}\left(3\left(\left(\alpha_1\frac{1-(\alpha_1 T_c)^K}{1-(\alpha_1 T_c)^{(K+1)}}\right)\left(\max\{0,(\overline{W_{Q_1}}-D_{Q_1})\}+\min\{\overline{W_{Q_1}},D_{Q_1}\}\right)\right)+1\right). \tag{22}$$

4.2.2. Average Queue Waiting Time in Q_2

The average queue waiting time $\overline{W_{Q_2}}$ of a tagged T_2 packet depends not only on the packets found in Q_1 upon arrival, but also the subsequent arrivals of T_1 packets. According to Little's formula [39], there are on average $\overline{\alpha_1}\overline{W_{Q_1}}$ packets found in Q_1; in addition, a number of T_1 packets may arrive while the tagged packet waits in Q_2, which is on average $\overline{\alpha_1}\overline{W_{Q_2}}$ packets. Since Q_2 follows the EDF approach to schedule T$_2$ packets, it follows the same scenario illustrated in Figure 6. Hence, $\overline{W_{Q_2}}$ can be expressed as follows:

$$\overline{W_{Q_2}} = T_c \left(\frac{3}{2} \left(\overline{\alpha_1} \overline{W_{Q_1}} + \overline{\alpha_1} \overline{W_{Q_2}} + \overline{N_2^B} + \overline{N_2^A} \right) + \frac{1}{2} \right). \tag{23}$$

Following the same procedures of Equations (13)–(21), $\overline{W_{Q_2}}$ is given as:

$$\begin{aligned} \overline{W_{Q_2}} = \frac{T_c}{2} \Bigg(3 \Bigg(\left(\alpha_1 \frac{1-(\alpha_1 T_c)^K}{1-(\alpha_1 T_c)^{(K+1)}} \right) \left(\overline{W_{Q_1}} + \overline{W_{Q_2}} \right) \\ + \left(\alpha_2 \frac{1-(\alpha_2 T_c)^K}{1-(\alpha_2 T_c)^{(K+1)}} \right) \left(\max\left\{0, \left(\overline{W_{Q_2}} - D_{Q_2}\right)\right\} + \min\left\{\overline{W_{Q_2}}, D_{Q_2}\right\} \right) \Bigg) + 1 \Bigg). \end{aligned} \tag{24}$$

4.2.3. Average Queue Waiting Time in Q$_3$

According to the FIFO approach in Q$_3$, an arbitrary tagged T$_3$ packet has to wait for the transmission of all packets that arrived ahead. Since Q$_3$ has the lowest priority, $\overline{W_{Q_3}}$ is directly affected by the arrivals of T$_1$ and T$_2$. In addition to the average number of packets already waiting in Q$_1$, Q$_2$, and Q$_3$, which are $\overline{\alpha_1}\overline{W_{Q_1}}$, $\overline{\alpha_2}\overline{W_{Q_2}}$, and $\overline{\alpha_3}\overline{W_{Q_3}}$, respectively, a tagged packet that arrives at Q$_3$ has to wait for higher priority packets to arrive at Q$_1$ and Q$_2$, which are on average $\overline{\alpha_1}\overline{W_{Q_3}} + \overline{\alpha_2}\overline{W_{Q_3}}$ packets. Accordingly, $\overline{W_{Q_3}}$ is given as:

$$\begin{aligned} \overline{W_{Q_3}} &= \frac{T_c}{2} \left(3 \left(\overline{\alpha_1}\overline{W_{Q_1}} + \overline{\alpha_2}\overline{W_{Q_2}} + \overline{\alpha_3}\overline{W_{Q_3}} + \overline{\alpha_1}\overline{W_{Q_3}} + \overline{\alpha_2}\overline{W_{Q_3}} \right) + 1 \right) \\ &= \frac{T_c}{2} \Bigg(3 \Bigg(\left(\alpha_1 \frac{1-(\alpha_1 T_c)^K}{1-(\alpha_1 T_c)^{(K+1)}} \right) \left(\overline{W_{Q_1}} + \overline{W_{Q_3}} \right) + \left(\alpha_2 \frac{1-(\alpha_2 T_c)^K}{1-(\alpha_2 T_c)^{(K+1)}} \right) \left(\overline{W_{Q_2}} + \overline{W_{Q_3}} \right) \\ &+ \left(\alpha_3 \frac{1-(\alpha_3 T_c)^K}{1-(\alpha_3 T_c)^{(K+1)}} \right) \overline{W_{Q_3}} \Bigg) + 1 \Bigg). \end{aligned} \tag{25}$$

The average queue waiting times given in Equations (22), (24), and (25) must satisfy the Kleinrock conservation law [40]:

$$\sum_{i=1}^{3} \rho_i \overline{W_{Q_i}} = \frac{\sum_{i=1}^{3} \rho_i \overline{W_0}}{1 - \sum_{i=1}^{3} \rho_i}, \tag{26}$$

where $\rho_i = \overline{S_i}\overline{\alpha_i}$ is the utilization factor of Q$_i$, $\overline{S_i}$ is the average service time of packets in Q$_i$, and $\overline{W_0}$ is the average residual service time of the packet whose transmission is in progress. According to the mean residual life formula [40], $\overline{W_0}$ is given as:

$$\overline{W_0} = \sum_{i=1}^{3} \rho_i \frac{\overline{S_i^2}}{2\overline{S_i}} = \frac{T_c^2}{2} \sum_{i=1}^{3} \overline{\alpha_i}. \tag{27}$$

Since we have a deterministic service time for Q$_1$, Q$_2$ and Q$_3$, i.e., $\overline{S_i^2} = T_c^2$, $\overline{W_0}$ in Equation (27) can be obtained as follows:

$$\overline{W_0} = \frac{T_c^2}{2} \left(\alpha_1 \frac{1-(\alpha_1 T_c)^K}{1-(\alpha_1 T_c)^{(K+1)}} + \alpha_2 \frac{1-(\alpha_2 T_c)^K}{1-(\alpha_2 T_c)^{(K+1)}} + \alpha_3 \frac{1-(\alpha_3 T_c)^K}{1-(\alpha_3 T_c)^{(K+1)}} \right) \tag{28}$$

Having deterministic values for T_c, D_{Q_1}, and D_{Q_2}, the average queue waiting time $\overline{W_{Q_i}}$ of each queue can be solved via the set of non-linear Equations (22), (24), and (25) along with the conservation formula given by Equation (26).

5. Performance Evaluations

We evaluated the performance of our proposed work and compared it with existing work under different performance metrics. The results were obtained through extensive Monte Carlo simulations in MATLAB. We used the parameters defined in Table 1 to model the simulation environment as close to the real conditions as possible.

Table 1. Simulation parameters.

Parameter	Value
Network size	30 nodes
Propagation model	Shadowing (log-normal)
Standard deviation	14 dB
Deployment area	200 m $\times$ 200 m
Transmission range	30 m
Data rate	250 kB/s
Packet length	100 B
Slotframe length	200 slots
Time slot duration	10 ms
No. of channels	4
Output buffer size	10 packets
No. of retransmissions	3
I_{min}	3 s
Δ	0.25
θ	0.5
δ	0.5
m	4
α_1	1/50 s
α_2	1/20 s

We considered a network of 30 nodes that were randomly deployed in a 200 m $\times$ 200 m area. Communications were carried out through a predefined TSCH schedule. Constructing and maintaining the TSCH schedule were out of scope of this paper. We considered log-normal shadowing distribution for the channel model with the specified standard deviation selected according to the measurements reported in [41]. Each of the following results were averaged over 10 simulation runs with each lasting for a duration of 1000 consecutive slotframes. The results were produced with a 95% confidence interval based on the t-distribution. In the following text and figures, we refer to our proposed Congestion Control and Traffic Differentiation method as CCTD. We evaluated the performance of CCTD under two scenarios. The first scenario considered a single-traffic network model where nodes generated only T_3 traffic periodically according to a specific rate. The second scenario considered a multi-traffic network model where each node generated both T_1 and T_2 packets according to a Poisson process with parameters α_1 and α_2, respectively, as defined in Table 1, along with the periodic T_3 packets. For both scenarios, we considered a fixed packet size of 100 B for all traffic types [42]. Since timeliness and reliability were of primary importance to support IIoT applications, our results mainly focused on the Packet Delivery Ratio (PDR) and End-to-End (E2E) delay parameters.

5.1. Single-Traffic Scenario

First, we describe the effect of the proposed congestion control method on the network topology by sketching the DODAG created by RPL-OF0 [8] and CCTD in Figure 8a,b, respectively. Each DODAG structure in Figure 8 represents the routing topology recognized at the end of the simulations by tracking each child-parent pair in each end-to-end path through the node ID. As shown in the figure, the DODAG created by RPL-OF0 suffered from imbalanced load distribution among parent nodes. For instance, Node 21 had the burden to forward almost 43% of the total load of the network through its corresponding sub-tree, which was much less than other nodes with the same RANK, e.g., Nodes 20, 23, and 25. The situation was even worse for Node 28, which had the responsibility to forward the traffic of almost 70% of the network. This imbalanced network was mainly due to the adopted parent selection mechanism in RPL-OF0, which was unaware of the congestion situation at parent nodes, and hence imposed significant degradation in the network performance in terms of delay and packet delivery. However, the proposed CCTD distributed the traffic load fairly among intermediate nodes, as shown in Figure 8b. The CCTD approach managed to reduce the standard deviation of the number of children per node from 1.6 to 0.73. This was due to the adopted congestion-aware parent selection mechanism in CCTD where a node selected its preferred parent according to its queue occupancy, which was updated efficiently through the improved Trickle timer algorithm.

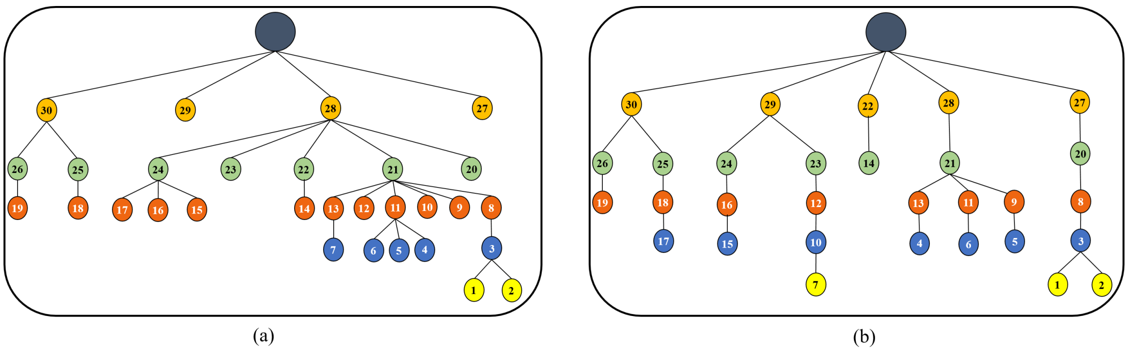


Figure 8. DODAG created by: (**a**) RPL-Objective Function (OF) 0; (**b**) Congestion Control and Traffic Differentiation (CCTD).

Next, we evaluated the impact of the design parameters λ and Γ on the performance of the proposed CCTD. Figure 9 shows the PDR at a packet generation rate of 90 packets per minute (ppm). The PDR was calculated by dividing the number of packets successfully received at the DODAG root by the total packets generated in the network. We first observed that the PDR improved as λ increased due to the fact that increasing λ in Equation (7) made the parent switching mechanism mainly based on the queue backlog factor to avoid congestion, hence improving the PDR. However, the PDR then decreased until it stayed almost constant as larger values of λ may cause the node to select parents with longer paths and/or unreliable links. Furthermore, we observed that the PDR performance almost followed the same trend with varying Γ. For large values of Γ, the nodes had a high tendency to change their parents as soon as a congestion was detected, which constituted the trade-off between fast load balance and the thundering herd effect as mentioned in Section 3. Therefore, the values of λ and Γ could be empirically adjusted by observing network performance. Based on Figure 9, we selected $\lambda = 4$ and $\Gamma = 0.5$ for the following results as these values gave the best PDR performance.

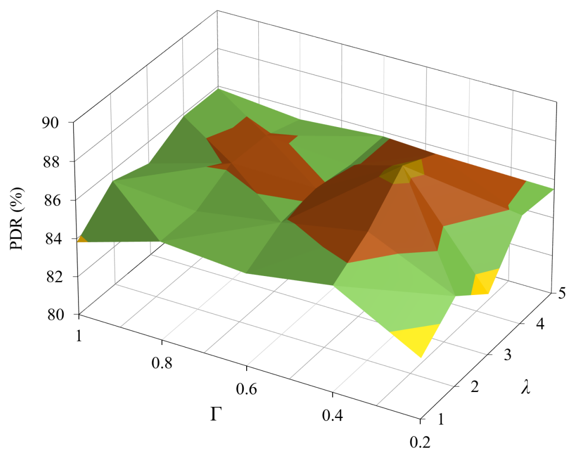


Figure 9. PDR performance under different values of λ and Γ at a traffic load of 90 ppm/node.

Hereafter, we compare the proposed work with RPL-OF0 and CoAR [30] under different performance metrics. Figure 10 shows the Queue Loss Ratio (QLR) of the three schemes for different traffic rates. The QLR was calculated as the total packets lost due to queue overflow divided by the total generated packets in the network. Although queue losses and buffer overflow are inevitable under heavy load conditions, efficient load-balancing and congestion control could mitigate such problems. The proposed CCTD approach managed to reduce the QLR of RPL-OF0 by 79% at 150 ppm/node as a result of the adopted congestion control framework, which helped achieve a fair load distribution among intermediate nodes. On the other hand, the proposed probabilistic parent selection strategy in Equation (10) and the improved Trickle timer algorithm together helped to improve the QLR performance compared to that achieved by CoAR. For instance, CCTD improved the QLR of CoAR by 45% at 150 ppm/node, which was increased to 53% at 180 ppm/node.

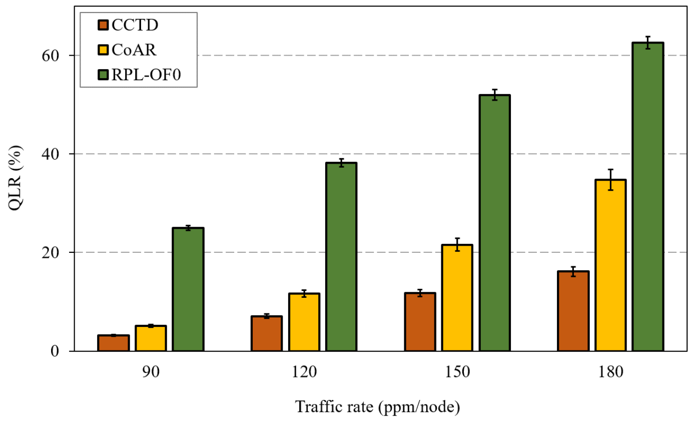


Figure 10. Queue Loss Ratio (QLR) comparison for different traffic rates.

As mentioned earlier, RPL was implemented on LLN resource-constrained nodes with small queue sizes, and these queues started to overflow under heavy traffic load. A solution could be to increase the buffer size of intermediate nodes to alleviate the congestion problem. However, such a solution was ineffective in the case of imbalanced RPL networks without a proper parent selection mechanism. To further illustrate, Figure 11 shows the QLR performance of RPL-OF0 after increasing the Buffer Size, denoted as BS in the figure, to 20 and 40. For the sake of comparison, we also added the QLR of CCTD with the BS of 10 packets. As depicted by Figure 11, increasing the buffer size to the double value marginally improved the QLR in RPL-OF0 under heavy traffic conditions. For instance, the QLR was reduced by 9% when increasing the BS from 20 to 40 at 120 ppm/node, while it was reduced to only 3% at a traffic rate of 180 ppm/node. On the other hand, CCTD with a BS of 10 maintained improved QLR performance over RPL-OF0 under heavy traffic rates. In order to further prove the effectiveness of the proposed method, Figure 11 shows that at a rate of 150 ppm/node, RPL-OF0 could achieve almost the same QLR as CCTD when increasing the BS to 220. Such a queue size is, however, infeasible in practice given the current resource-constrained LLN devices.

As the QLR reduced, the proposed CCTD in turn enhanced the PDR performance compared to RPL-OF0 and CoAR, as shown in Figure 12. The effectiveness of the proposed CCTD was clearly observed at heavy load conditions where CCTD improved the performance of CoAR by 33% and 59% at 150 ppm/node and 180 ppm/node, respectively.

Figure 13 shows the hop-count comparison between the three methods against different traffic rates. The figure includes both the average and the maximum hop-count to the DODAG root. As can be noted from the figure, the proposed CCTD had a marginal effect on the hop-count of RPL-OF0 as nodes may select a path with a higher hop distance towards the DODAG root to alleviate the congestion effect. Moreover, CCTD exploited the hop-count metric in $R^{LB}(p_i)$ as shown by (7) when selecting the alternative parent; therefore, CCTD incurred a slight increase in this metric. Furthermore, the parameter λ could be further tuned to achieve a trade-off between congestion control and hop-count. However, since CoAR completely excluded the hop-count from the routing metric in its congestion control mechanism, it showed a higher hop-count than that of CCTD and RPL-OF0.

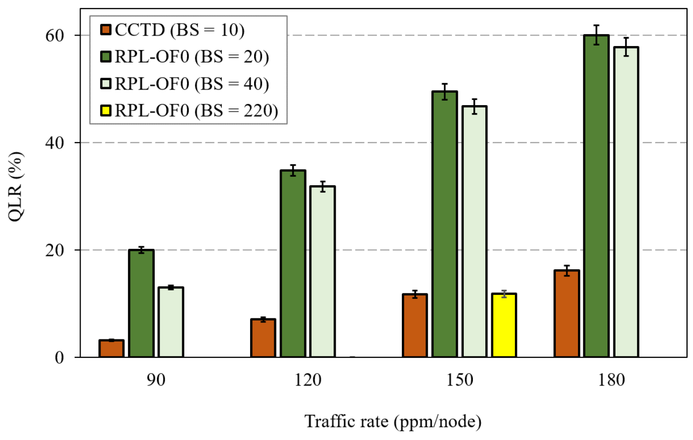


Figure 11. Effect of the buffer size increase on QLR.

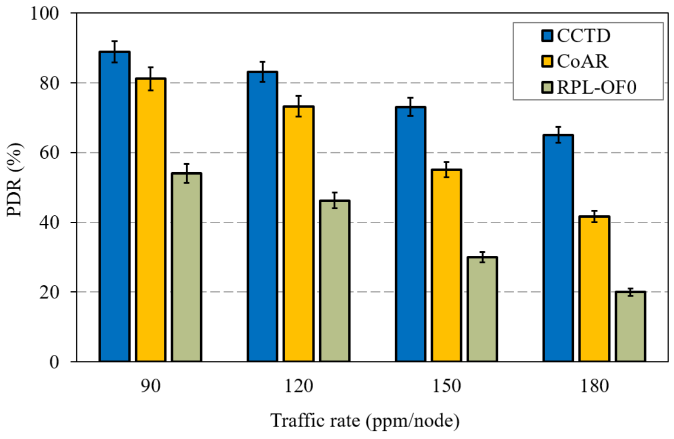


Figure 12. PDR performance comparison for different traffic rates.

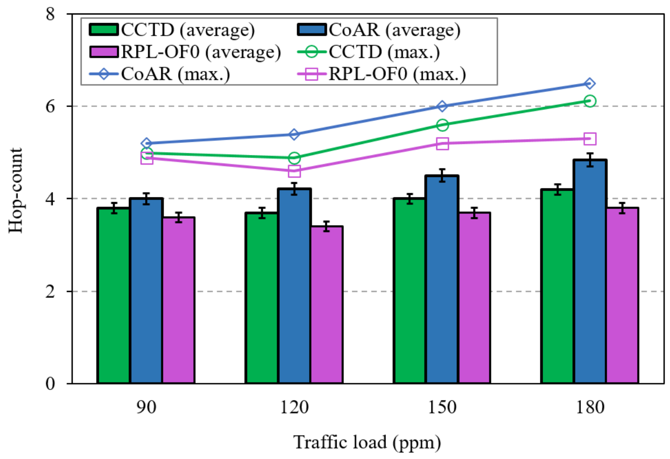


Figure 13. Hop-count comparison for different traffic rates.

For a quick look at the effect of the network size on the performance of the proposed CCTD, Figure 14 shows a PDR comparison of the three schemes with a varying number of nodes in the network. First, we observed that the PDR performance of RPL-OF0 significantly degraded as the network size increased. This was because the network was at a great risk of suffering from an imbalanced load distribution in terms of DODAG construction, which caused severe queue losses.

Hence, congestion control and load balancing become more important and an effective way to improve the performance of RPL in large-scale networks. Accordingly, as shown in Figure 14, CCTD maintained its PDR performance enhancement and even achieved more improvements as the network size increases. Specifically, CCTD improved the PDR by 64% compared to RPL-OF0 with 30 nodes, while the percentage was dramatically increased to 275% with 150 nodes. In this context, it is reported that in process automation scenarios, the network would include a maximum of 50 nodes in order to meet the required refresh rates [43].

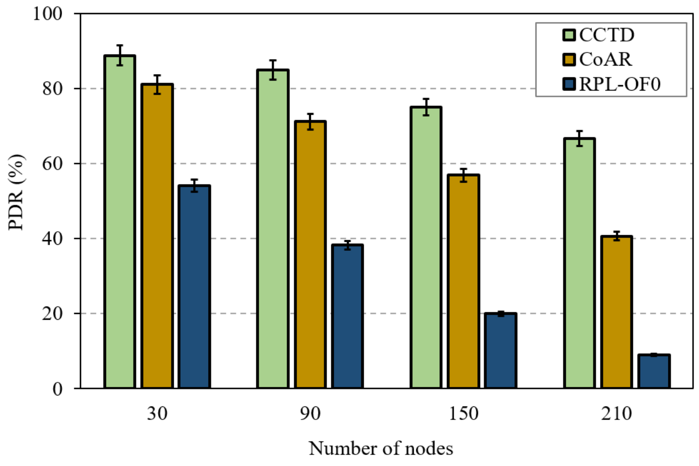


Figure 14. PDR performance comparison for different network sizes with a traffic rate of 90 ppm/node.

5.2. Multi-Traffic Scenario

Hereafter, we consider the heterogeneous network model to emphasize the effect of the proposed multi-queue model in CCTD.

Figure 15a,b shows the worst-case E2E comparison of T_1 and T_2 traffic, respectively, under different traffic rates of T_3. The E2E delay was calculated as the time elapsed from the generation of the packet until it was successfully delivered to the DODAG root. As shown in the two figures, the proposed CCTD scheme significantly reduced the worst-case delay of critical traffic T_1 and T_2 compared to RPL-OF0 and CoAR. With the proposed priority-based multi-queue model, critical packets were always given higher priority for transmission as long as Q_1 and Q_2 were non-empty where packets from Q_1 were given the highest priority. Such a function was not provided either by RPL-OF0 or CoAR, where packet transmissions followed an FIFO-based single-queue model regardless of the priority and the criticality of the traffic. Hence, critical data were likely to be blocked by the transmission of multiple non-critical data as the latter arrived first. The situation became even worse at high traffic rates of T_3 as more traffic was queued ahead to the critical data, which further degraded its delay performance, while CCTD preserved a steady lower delay regardless of the T_3 traffic load as shown in Figure 15a,b. Moreover, buffer overflow was likely to occur in RPL-OF0 and CoAR with the single queue model in heavy traffic scenarios where T_1 and T_2 packets were dropped and needed to be retransmitted, which added significantly to the delay. Quantitatively, compared to CoAR, CCTD reduced the worst-case delay of T_1 by at least 78% while it achieved a 68% reduction for T_2.

The introduced service differentiation was further illustrated through the Cumulative Distribution Function (CDF) comparison of E2E delay as shown by Figure 16a,b where we plot the CDF at 120 ppm/node in CCTD and CoAR, respectively. Figure 16a clearly shows the service differentiation between the different traffic types in CCTD as a result of the proposed multi-queue model where T_1 packets in Q_1 were given the highest priority priority, while T_3 packets were given the lowest priority. However, Figure 16b shows that all traffic in CoAR almost followed the same trend for the E2E delay distribution. In CoAR, the packet in the output buffer followed the FIFO scheduling policy; hence, on average, all packets experienced identical E2E delay.

The results in Figure 16a also reflected on-time PDR of both T_1 and T_2. The on-time PDR denotes the percentage of packets delivered within a predefined deadline. As mentioned earlier, for critical traffic, out of the total packets delivered successfully to the DODAG root, those delivered within the specified deadline limit were meaningful to the system. For instance, CCTD achieved a one-time PDR of 86% considering a deadline of 400 ms for T_1, while the percentage was 92% considering a deadline of 500 ms for T_2.

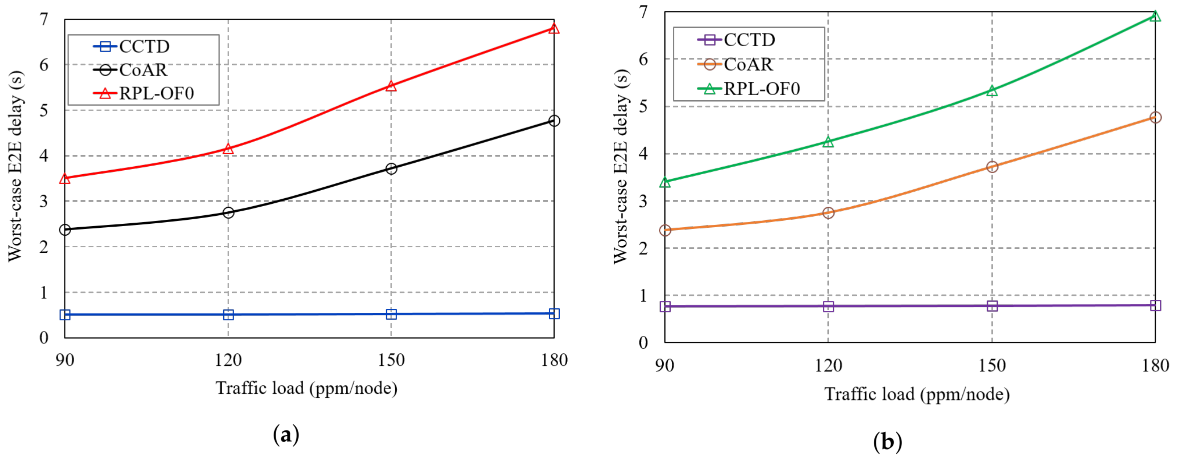


Figure 15. Worst-case delay comparison of: (**a**) T_1 traffic; (**b**) T_2 traffic.

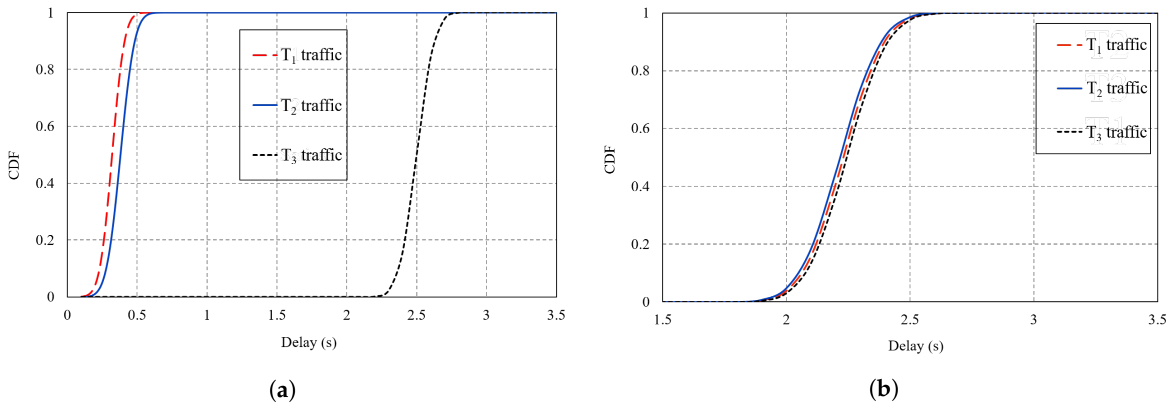


Figure 16. CDF of the E2E delay of all traffic types at 120 ppm/node: (**a**) CCTD; (**b**) Congestion-Aware Routing (CoAR).

The introduced multi-queue model in CCTD had a direct effect on the delivery performance of T_3, as the buffered T_3 packets in Q_3 were suspended from transmission as long as Q_1 and/or Q_2 were non-empty. Hence, CCTD sacrificed the reduced PDR performance of T_3 in favor of the improved performance for T_1 and T_2. Figure 17 shows the PDR comparison of T_3 for different traffic rates. CCTD had a reduction of 32% in the PDR of T_3 at 180 ppm/node compared to CoAR, which was relaxed to 13% at 90 ppm/node. However, such degradation could be considered insignificant in critical IIoT applications where guaranteeing real-time performance of critical data is a primary priority [17]. In addition, T_1 and T_2 occurred infrequently, and such degradation was not persistent in the network. However, CCTD preserved improved PDR performance against RPL-OF0 due to the proposed congestion control mechanism, which greatly compensated the degradation caused by the multi-queue model.

Finally, we investigated the routing overhead introduced by the threes schemes in terms of average number of transmitted DIO messages. In this context, Figure 18 shows the average DIO messages transmitted per node per hour. As shown in the figure, both CCTD and CoAR incurred additional DIO overhead compared to that in RPL-OF0, especially under heavy traffic load. This was because nodes in CCTD and CoAR transmitted more DIO messages to distribute the congestion information based on the corresponding Trickle timer reset strategy in each scheme. The increase in the overhead, however, could be considered a reasonable cost to achieve a fair load distribution in the network and avoid the severe consequences of network congestion. Furthermore, the additional overhead was generally insignificant compared to the total active traffic in the network. For instance, at 90 ppm/node, the overhead in CCTD represented less than 1% of the total traffic.

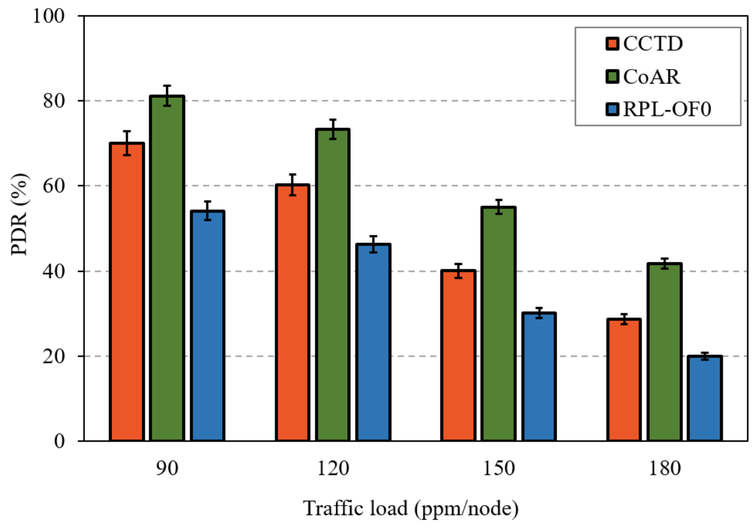


Figure 17. PDR comparison of T_3 for different traffic rates.

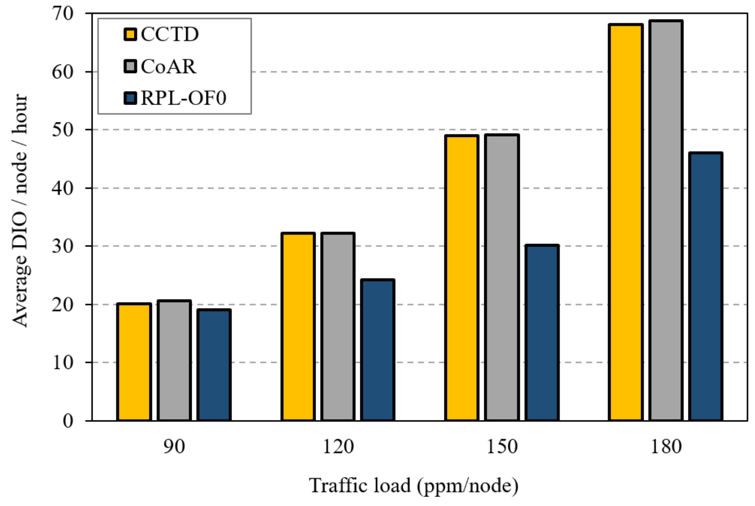


Figure 18. Average number of DODAG Information Object (DIO) messages for different traffic rates of T_3.

6. Conclusions and Future Work

In this paper, we proposed a congestion control and service differentiation strategy to support heterogeneous 6TiSCH networks in IIoT applications. The congestion was monitored through sharing the backlog information among neighbor nodes, which was implicitly embedded in the DIO message. The preferred parent of each node was selected based on its queue occupancy to achieve a fair load distribution in the network. A modified Trickle timer strategy was also introduced to detect and act on congestion in timely manner while keeping overhead to a minimum. Moreover, we proposed a multi-queue model to support prioritized and low delay transmission of critical traffic. In addition, we provided a mathematical analysis of the average waiting time of each priority queue. Performance evaluations were carried out, and the results proved the effectiveness of our method to achieve improved packet delivery and low queue losses under heavy load scenarios, as well as the improved delay performance of critical traffic with an insignificant increase in the overhead.

As future work, the performance of the proposed framework can be further investigated in terms of power consumption and network lifetime metrics, as well as providing mathematical models for the throughput and worst-case delay of each traffic category. Moreover, we plan to have a full implementation of the proposed framework to evaluate its performance in real scenarios. Optimizing the TSCH schedule to further improve the channel access and transmission delay of critical data is also left as a future investigation.

Author Contributions: H.F. conceived the proposed protocol framework, produced the mathematical model and the simulation results, and wrote the paper. P.Ö. and M.G. supervised the overall work and revised the manuscript. All authors read and agreed to the published version of the manuscript.

Funding: This work was supported by the Swedish Knowledge Foundation (KKS) (Grant No. 2018170) and the Swedish Foundation for Strategic Research (SSF) (Grant No. SM18-0044).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sisinni, E.; Saifullah, A.; Han, S.; Jennehag, U.; Gidlund, M. Industrial internet of things: Challenges, opportunities, and directions. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4724–4734. [CrossRef]
2. Xu, H.; Yu, W.; Griffith, D.; Golmie, N. A Survey on industrial internet of things: A cyber-physical systems perspective. *IEEE Access* **2018**, *6*, 78238–78259. [CrossRef]
3. Vallati, C.; Brienza, S.; Anastasi, G.; Das, S.K. Improving network formation in 6tisch networks. *IEEE Trans. Mobile Comput.* **2019**, *18*, 98–110. [CrossRef]
4. Vitturi, S.; Zunino, C.; Sauter, T. Industrial communication systems and their future challenges: Next-generation ethernet, iiot, and 5g. *Proc. IEEE* **2019**, *107*, 944–961. [CrossRef]
5. Karaagac, A.; Haxhibeqiri, J.; Moerman, I.; Hoebeke, J. Time-critical communication in 6TiSCH networks. In Proceedings of the 2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), Barcelona, Spain, 15–18 April 2018; pp. 161–166.
6. Winter, T.; Thubert, P.; Brandt, A.; Hui, J.; Kelsey, R.; Levis, P.; Pister, K.; Struik, R.; Vasseur, J.; Alexander, R. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. Available online: https://tools.ietf.org/html/rfc6550 (accessed on 11 May 2020).
7. Ghaleb, B.; Al-Dubai, A.Y.; Ekonomou, E.; Alsarhan, A.; Nasser, Y.; Mackenzie, L.M.; Boukerche, A. A survey of limitations and enhancements of the Ipv6 routing protocol for low-power and lossy networks: A focus on core operations. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 1607–1635. [CrossRef]
8. Thubert, P. Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL). Available online: https:tools.ietf.org/html/rfc6552 (accessed on 11 May 2020).
9. Gnawali, O.; Levis, P. The Minimum Rank with Hysteresis Objective Function. Available online: https://tools.ietf.org/html/rfc6719 (accessed on 11 May 2010).
10. Levis, P.; Clausen, T.; Hui, J.; Gnawali, O.; Ko, J. The Trickle Algorithm. Available online: https://tools.ietf.org/html/rfc6206 (accessed on 20 May 2020).
11. Lim, C. A survey on congestion control for RPL-based wireless sensor networks. *Sensors* **2019**, *19*, 2567. [CrossRef] [PubMed]
12. Farag, H.; Sisinni, E.; Gidlund, M.; Österberg, P. Priority-aware wireless fieldbus protocol for mixed-criticality industrial wireless sensor networks. *IEEE Sens. J.* **2019**, *19*, 2767–2780. [CrossRef]
13. Kim, H.; Im, H.; Lee, M.; Paek, J.; Bahk, S. A measurement study of TCP over RPL in low-power and lossy networks. *J. Commun. Netw.* **2015**, *17*, 647–655. [CrossRef]
14. Tahir, Y.; Yang, S.; McCann, J. BRPL: Backpressure RPL for high-throughput and mobile IoTs. *IEEE Trans. Mob. Comput.* **2018**, *17*, 29–43. [CrossRef]
15. Capone, S.; Brama, R.; Accettura, N.; Striccoli, D.; Boggia, G. An energy efficient and reliable composite metric for RPL organized networks. In Proceedings of the 2014 12th IEEE International Conference on Embedded and Ubiquitous Computing, Milano, Italy, 26–28 August 2014; pp. 178–184.

16. Zhao, M.; Ho, I.W.; Chong, P.H.J. An energy-efficient region-based RPL routing protocol for low-power and lossy networks. *IEEE Internet Things J.* **2016**, *3*, 1319–1333. [CrossRef]
17. Farag, H.; Gidlund, M.; Österberg, P. A delay-bounded MAC protocol for mission- and time-critical applications in industrial wireless sensor networks. *IEEE Sens. J.* **2018**, *18*, 2607–2616. [CrossRef]
18. Shen, W.; Zhang, T.; Barac, F.; Gidlund, M. PriorityMAC: A priority- enhanced MAC protocol for critical traffic in industrial wireless sensor and actuator networks. *IEEE Trans. Ind. Inform.* **2014**, *10*, 824–835. [CrossRef]
19. Zheng, T.; Gidlund, M.; Åkerberg, J. WirArb: A new mac protocol for time critical industrial wireless sensor network applications. *IEEE Sens. J.* **2016**, *16*, 2127–2139. [CrossRef]
20. Lall, S.; Alfa, A.S.; Maharaj, B.T. The role of queuing theory in the design and analysis of wireless sensor networks: An insight. In Proceedings of the 2016 IEEE 14th International Conference on Industrial Informatics (INDIN), Poitiers, France, 19–21 July 2016; pp. 1191–1194.
21. Qiu, T.; Zheng, K.; Han, M.; Chen, C.L.P.; Xu, M. A data-emergency-aware scheduling scheme for internet of things in smart cities. *IEEE Trans. Ind. Inform.* **2018**, *14*, 2042–2051. [CrossRef]
22. Nasser, N.; Karim, L.; Taleb, T. Dynamic multilevel priority packet scheduling scheme for wireless sensor network. *IEEE Trans. Wirel. Commun.* **2013**, *12*, 1448–1459. [CrossRef]
23. Lee, E.M.; Kashif, A.; Lee, D.H.; Kim, I.T.; Park, M.S. Location based multi-queue scheduler in wireless sensor network. In Proceedings of the 2010 The 12th International Conference on Advanced Communication Technology (ICACT), Phoenix Park, Korea, 7–10 February 2010; Volume 1, pp. 551–555.
24. Bhandari, S.; Sharma, S.K.; Wang, X. Latency minimization in wireless IoT using prioritized channel access and data aggregation. In Proceedings of the GLOBECOM 2017—2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–6.
25. Ha, M.; Kwon, K.; Kim, D.; Kong, P. Dynamic and distributed load balancing scheme in multi-gateway based 6LoWPAN. In Proceedings of the 2014 IEEE International Conference on Internet of Things (iThings), Taipei, Taiwan, 1–3 Septembar 2014; pp. 87–94.
26. Al-Kashoash, H.A.A.; Hafeez, M.; Kemp, A.H. Congestion control for 6LoWPAN networks: A game theoretic framework. *IEEE Internet Things J.* **2017**, *4*, 760–771. [CrossRef]
27. Al-Kashoash, H.A.A.; Hafeez, M.; Kemp, A.H. Congestion control in wireless sensor networks by hybrid multi-objective optimization algorithm. *Comput. Netw.* **2018**, *138*, 90–107.
28. Al-Kashoash, H.A.A.; Amer, H.M.; Mihaylova, L.; Kemp, A.H. Optimization-based hybrid congestion alleviation for 6LoWPAN networks. *IEEE Internet Things J.* **2017**, *4*, 2070–2081. [CrossRef]
29. Lodhi, M.A.; Rehman, A.; Khan, M.M.; Hussain, F.B. Multiple path RPL for low power lossy networks. In Proceedings of the 2015 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob), Bandung, Indonesia, 27–29 August 2015; pp. 279–284.
30. Bhandari, K.S.; Hosen, A.S.M.S.; Cho, G.H. CoAR: Congestion-aware routing protocol for low power and lossy networks for IoT applications. *Sensors* **2018**, *18*, 3838. [CrossRef] [PubMed]
31. Coladon, T.; Vučinić, M.; Tourancheau, B. Multiple redundancy constants with Trickle. In Proceedings of the Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, Hong Kong, China, 30 August–2 September 2015; pp. 1951–1956.
32. Vučinić, M.; Król, M.; Jonglez, B.; Coladon, T.; Tourancheau, B. Trickle-D: High fairness and low transmission load with dynamic redundancy. *IEEE Internet Things J.* **2017**, *4*, 1477–1488. [CrossRef]
33. Vasseur, J.; Kim, M.; Pister, K.; Dejean, N.; Barthel, D. Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks. Available online: https://tools.ietf.org/html/rfc6551 (accessed on 20 May 2020).
34. Hou, J.; Rahul, J.; Luo, Z. Optimization of Parent-Node Selection in RPL-Based Networks. Available online: https://www.ietf.org/archive/id/draft-hou-roll-RPL-parent-selection-00.t (accessed on 20 May 2020).
35. Giles, H.F.; Wagner, G.R.; Mount, E.M. *Extrusion: The Definitive Processing Guide and Handbook*; William Andrew: Oxford, UK, 2014.
36. Rauwendaal, C. *Understanding Extrusion*; Hanser: Cincinnati, OH, USA, 2010.
37. Buttazzo, G.C. *Hard rEal-Time Computing Systems: Predictable Scheduling Algorithms and Applications*; Springer: New York, NY, USA, 2011.
38. Rom, R.; Sidi, M. *Multiple Access protocols: Performance and Analysis*; Springer: New York, NY, USA, 1989.
39. Kleinrock, L. *Queueing Systems Volume 1: Theory*; Wiley: Hoboken, NJ, USA, 1975.

40. Kleinrock, L. *Queueing Systems Volume 2: Computer Applications*; Wiley: Hoboken, NJ, USA, 1976.
41. Miranda, J.; Abrishambaf, R.; Gomes, T.; Gonçalves, P.; Cabral, J.; Tavares, A.; Monteiro, J. Path loss exponent analysis in Wireless Sensor Networks: Experimental evaluation. In Proceedings of the 2013 11th IEEE International Conference on Industrial Informatics (INDIN), Bochum, Germany, 29–31 July 2013; pp. 54–58.
42. Sisinni, E.; Tramarin, F. 9—Isochronous wireless communication system for industrial automation. In *Industrial Wireless Sensor Networks*; Woodhead Publishing: Cambridge, UK, 2016; pp. 167–188. [CrossRef]
43. Åkerberg, J.; Gidlund, M.; Björkman, M. Future research challenges in wireless sensor and actuator networks targeting industrial automation. In Proceedings of the 2011 9th IEEE International Conference on Industrial Informatics, Caparica, Lisbon, 26–29 July 2011; pp. 410–415.