DIO Messages and Trickle Timer analysis of RPL Routing Protocol for UAV-assisted Data Collection in IoT

Bishmita Hazarika
Rakesh Matam
me.bishmita@gmail.com
rakesh@iiitg.ac.in
Indian Institute of Information Technology Guwahati
Guwahati, Assam, India

Mithun Mukherjee
Guangdong University of Petrochemical Technology
Maoming, China
m.mukherjee@ieee.org

Varun G Menon
SCMS School of Engineering and Technology
Ernakulam, India
varunmenon@ieee.org

ABSTRACT

Routing protocol for low-power and lossy networks (RPL) is an widely-used IPv6 routing protocol for lossy wireless networks with the power constrained devices in Internet of Things (IoT). It is a proactive protocol that constructs a destination oriented directed acyclic graph (DODAG) rooted at the single destination called the root node that resides at unmanned aerial vehicle (UAV). Specifically, a DODAG is built with the help of different control messages like DODAG information object (DIO), DODAG advertisement object (DAO), and DODAG information solicitation (DIS). As the generation of these messages incur additional energy consumption, RPL uses the Trickle algorithm to dynamically adjust the transmission windows. In this paper, we analyze the effect of the two parameters, namely, `DIO-INTERVAL-MINIMUM` and `DIO-INTERVAL-DOUBLING` that have significant effect on the Trickle algorithm and the rate of message generation. Through experiments, we show that an optimal selection of these parameters saves a significant amount of energy with different parameter settings in UAV-assisted IoT networks.

CCS CONCEPTS

• **Computer systems organization → Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

Energy-efficiency, Optimization, Routing protocol, RPL, UAV, DODAG

ACM Reference Format:
Bishmita Hazarika, Rakesh Matam, Mithun Mukherjee, and Varun G Menon. 2020. DIO Messages and Trickle Timer analysis of RPL Routing Protocol for UAV-assisted Data Collection in IoT. In *DroneCom '20: ACM MobiCom 2020 Workshop on Drone Assisted Wireless Communications for 5G and Beyond, September 25, 2020, London, UK*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3414045.3415944

1 INTRODUCTION

With the recent advancements in embedded computing and wireless technologies, a variety of physical things are getting connected to the Internet. Typically, the devices are generally constrained in terms of processing power, memory, and energy (battery power), similar to wireless sensor networks (WSNs) and a network of such devices is called a low-power lossy network (LLN). The wireless links interconnecting these devices are characterized by high loss rates and low data rates. To enable communication among such devices, the routing over low-power and lossy network (ROLL) working group has specified the IPv6 routing protocol for LLN, called RPL [6, 10]. The devices running RPL are connected in a tree-like topology and the connections are established forming a destination-oriented directed acyclic (DODAG) graph, where all the nodes are directed towards a common root. These graphs are formed based on an objective function (OF) that defines the metrics and constraints for the nodes running RPL, and helps to compute routes from data generating devices to the root node.

As the devices are battery powered, energy conservation mechanisms becomes crucial to maximize the network lifetime. Although the design of `OF` is central to determining the energy spent for network execution, there are several other RPL parameters that play a significant role. RPL uses

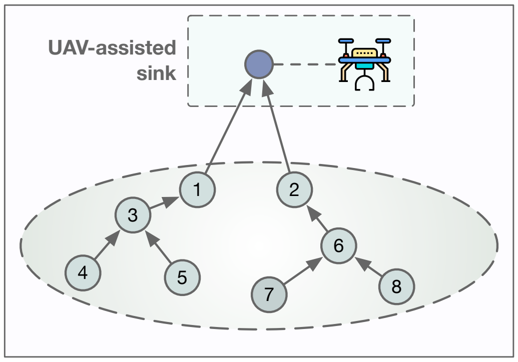

Figure 1: System model.

different control messages for topology formation, and frequent generation of such messages that results in higher energy costs. Unmanned aerial vehicle (UAV)-assisted networks have received significant attention in recent years due to the numerous advantages offered in real time surveillance and data collection applications. When deployed with an efficient architecture and well coordinated system, UAV systems that include drones, small air crafts, balloons, would provide reliable and cost-effective communication solutions for numerous real-world scenarios. With the unmanned aerial vehicle (UAV) as a root node, the task of data-collection becomes more challenging as additional control messages are required to maintain network connectivity. The pattern of generation of these control messages can be adapted within a network based on network behavior and requirements. In order to do so, it is important to understand and analyse these factors.

2 MOTIVATION AND CONTRIBUTION

Energy consumption is one of the critical issues in LLN. Different networks have different requirements and since RPL is the routing protocol designed to fulfill such requirements, it can be customised according to the network to save more energy and increase lifetime. To do so, understanding the factors within the protocol and how the protocol works is very important.

Therefore, in this paper, the role of parameters related to DODAG information object (DIO) control with a significant impact on the Trickle algorithm and the rate of message generation is analyzed. Through experiments, we show that an optimal selection of these parameters saves a significant amount of energy with different parameter settings for data collection in UAV-assisted IoT networks.

3 OVERVIEW OF RPL

The construction of network topology is initiated by the root node. In our system model, the UAVs are responsible to initiate the topology construction process. The DODAG is formed by nodes joining the graph and this is performed with the help of control messages by exchanging the information. There exists four types of control messages a) DODAG Information Object (DIO), b) Destination Advertisement Object (DAO), c) DODAG Information Solicitation (DIS), and d) and DAO Acknowledgement (DAO-ACK). An algorithm, called as *trickle Timer* [2] is used to suppress the generation of redundant control messages. Since our focus on the DIO message generation and Trickle Timer analysis [2, 4], we omit the discussion of other control messages for brevity. Alternatively, few works [3] have solely focused on optimizing the Trickle algorithm.

3.1 DIO Control Messages

To form the DODAG, the root node that resides on the UAV broadcasts the DIO messages. The DIO message mainly contains the following information: a) RPL instance ID, b) version number, c) rank, which is the relative position of a node from the root, d) Mode of Operation (MOP), root preferability (Prf), saves Sequence Number (DTSN), DODAG-ID etc.

All the nodes within the communication range of the root node receive the DIO message, when the root node multicasts the DIO message. Multiple roots might be sending DIO messages at the same time and hence a node might receive DIO messages from multiple roots having different OFs. A node cannot join more than one root at a time to form DODAG, so when a node receives multiple DIO messages, it runs an algorithm within itself to decide which root they want to join. The decision of the node is based upon the metrics defined by the OF of the root node. After a node joins the DODAG, if the node is not a leaf node, it further multicasts the DIO control messages so that other nodes can join the graph. These steps of forming the graph continues until no nodes are found within range or leaf nodes are found [1, 6–8, 8–10]

3.2 Trickle Timer Algorithm

For the maintainance and upgradation of topology, control messages need to be generated repeatedly, which consumes a lot of energy. To use the resources efficiently, generation of control messages is suppressed and are generated only when it is necessary. This is done by the Trickle Timer algorithm [5]. The Trickle timer algorithm uses a mechanism to keep a check on the consistency of packet generation pattern of the network. If the pattern is consistent, and does not have any redundant or old data then the Trickle timer reduces the rate of sending DIO control messages exponentially. But in case of any inconsistency in the network, the next DIO message is rescheduled and is sent at the latest time interval.

3.3 Parameters of Trickle Timer Algorithm

The parameters that govern the Trickle timer algorithm are: $I_{\min}$, which denotes the minimum time interval between two DIO messages. It is calculated as follows:

$$I_{\min} = 2^{\texttt{DIO-MINIMUM-INTERVAL}} . \tag{1}$$

$I_{\max}$ that represents the maximum time interval between two DIO messages. It is calculated as follows:

$$I_{\max} = I_{\min} \times 2^{\texttt{DIO-INTERVAL-DOUBLINGS}} . \tag{2}$$

The redundancy constant k represents the number of redundant messages. I is size of current time-interval, and the time interval within I is represented using t. Lastly, c denotes a counter that is used to control transmissions. The role of these parameters is discussed next in the Trickle timer algorithm.

3.4 Trickle Algorithm Rules

(1) In the first interval, I is set to any value with in the range $[I_{\min}, I_{\max}]$.
(2) Reset c to 0 and I is reset to a random point between $[I/2, I]$ at the beginning of an interval.
(3) Increment the counter c for every consistent transmission.
(4) Transmit a DIO message only if the counter c is less than the redundancy constant k, otherwise it is suppressed.
(5) When interval I expires, the timer doubles the interval until $I_{\max}$ is reached. After that, the new interval is again started as in Step 2.
(6) In case of an inconsistent transmission and if interval I is greater then $I_{\min}$ the interval timer is reset to $I_{\min}$ and a new interval starts as in Step 2. This is done even if the threshold is not reached.

4 PERFORMANCE EVALUATION

To optimize the performance of the protocol in accordance to the network requirements, we need to analyze the DIO message generation pattern, and the working of the Trickle timer algorithm. We carry out different simulations on the Cooja network simulator to evaluate the performance for various choices of parameters. Table: 1 summarizes the default simulation parameters. The performance of RPL for UAV-assisted data collection is performed by setting up a network of 16 randomly deployed nodes, with 15 senders (i.e., IoT devices) and one sink node with mobility (i.e., UAV). Fig. 2 illustrates the node deployment. The behavior of a randomly selected node (we considered node 10) is presented to understand the affect of these parameters on the the network.

As mentioned before, the parameters that determine the operation of the Trickle Timer are $I_{\min}$, $I_{\max}$, where $I_{\min}$ is based on value of `min` and $I_{\max}$ is dependent on the `doubling` factor. The RFC[10] recommends the default values of 12 and 8 for `min` and `doubling`, respectively. Also, the default value of k is set to 10. The values of $I_{\min}$ and $I_{\max}$ are calculated as follows.

Table 1: Simulation Parameters

Parameters	Values
OS	Contiki master version
Simulator	Cooja;
Radio Model	Unit Disk Graph Medium
OF	Expected Transmission Count (ETX)
Number of nodes	1 server, 15 clients
Transmission Reception ratio	100 percent
Sensor	Sky Mote
Simulation Time	1 Hour (approximately)

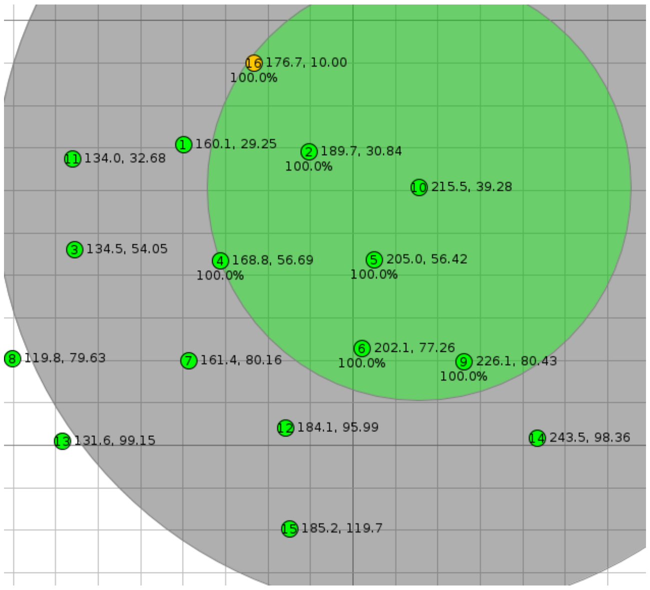

Figure 2: The Network set-up with the green circle representing the transmission range of node 10.

$$\begin{aligned} I_{\min} &= 2^{\texttt{min}} \\ &= 2^{12}\,\text{ms} = 4096\,\text{ms} = 4\,\text{s} \end{aligned} \tag{3}$$

$$\begin{aligned} I_{\max} &= I_{\min} \times 2^{\texttt{doubling}} \\ &= 4096 \times 2^{8}\,\text{ms} = 1\,048\,576\,\text{ms} = 17.5\,\text{min} . \end{aligned} \tag{4}$$

Thus, if the values of [`min`, `doubling`] are [12,8], then the minimum possible interval between two DIO messages is 5 s and maximum possible interval between two DIO messages becomes 17.5 min.

4.1 Delay analysis

The performance of the network is dependent on the parameter settings. To find the impact of these parameters, we

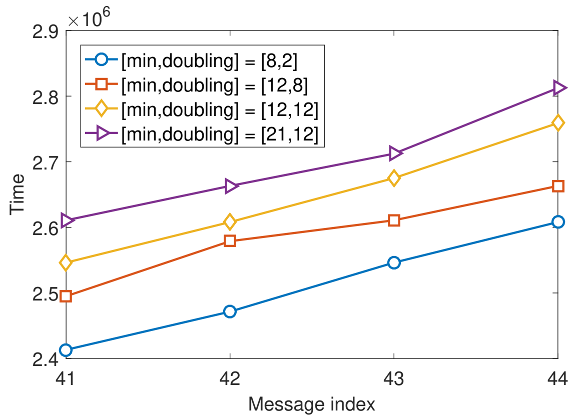

Figure 3: Total time taken (in millisecond) by each of the five cases to generate the last DIO message within the simulation time-interval.

compared the relative performance by considering five different pairs of values. For simplicity, we fixed the redundancy constant to $k = 10$ for all the scenarios. The DIO message generated only by the 'node 10' is considered for this study[1]. The five pair of values considered for the comparison are: 1) `min` = 12, `doubling` = 8, 2) `min` = 12, `doubling` = 12, 3) `min` = 21, `doubling` =2, 4) `min` = 21, `doubling` = 12, and 5)`min` = 8, doubling=2. For each of the above five cases, we run the simulation until the 45th DIO message packet is transmitted by the node 10. This allows us to verify for the pair of values for which the network takes the longest time to generate the 45th DIO message, and for what pair it take the least time. Considering a fixed time-interval, if fewer number of DIO messages are generated for a given pair of parameter settings, then the energy consumption would be relatively low in comparison to other scenarios.

Fig. 3 presents a comparison among all the five cases. We can see that for a given value of k=10, the value pair [`min`, `doubling`] = [12,12] takes the longest time to generate the 45th DIO message packet while [`min`, `doubling`] = [8,2] required the least time. This means that for values [12,12] the least number of DIO messages are generated per unit time, and hence it uses less energy, while value pair [8,2] generates more DIO messages per unit time and hence expend more energy. It is important to note that, among the five pairs of values considered, the pair [21,12] is the largest value pair and is expected to generates least DIO messages. This is because the interval between messages is longer but it takes less time than the pair [12,12]. This is because of the considered k value. As mentioned in the algorithm *Step4*, if the value of counter c becomes greater then the redundancy constant k, then the message generation is suppressed. Here, although the `min` was much greater, i.e., `min` = 21, the interval could not increment to the possible threshold as the messages are suppressed when the counter reaches $k = 10$ and a new interval is started.

[1]RPL-DIO-INTERVAL-MIN is referred as 'min' and RPL-DIO-INTERVAL-DOUBLINGS is referred as 'doubling' in short

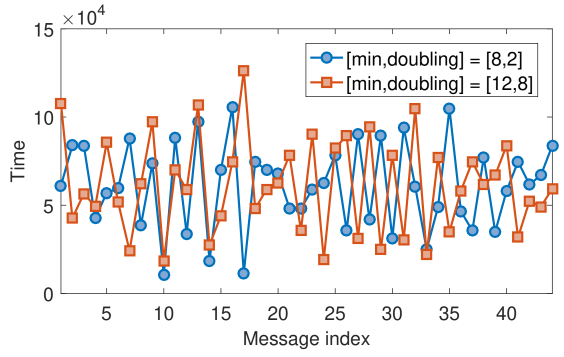

Figure 4: Performance comparison in terms of time taken for generation of DIO message for [min,doubling]=[8,2] and [12,8].

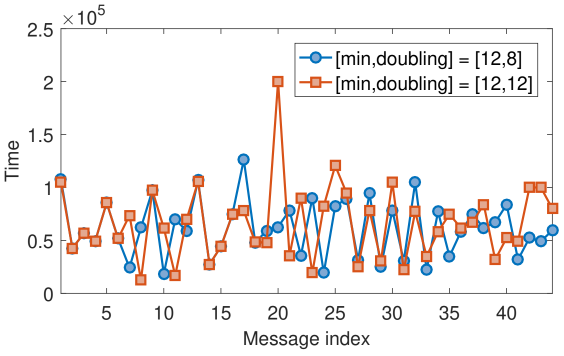

Figure 5: Performance comparison in terms of time taken for generation of DIO message for [min,doubling]=[12,8] and [12,12].

4.2 Time interval

Considering the same five pairs of values as mentioned above, we compare the time interval set by the Trickle timer to generate the DIO messages. The message generation pattern for the values [`min`, `doubling`]= [8,2], [12,12], [21,2], and [21,12] is compared to the default values as specified in the RFC 6550[10] , i.e., [`min`, `doubling`]=[12,8]. This helps us to practically understand how the trickle timer algorithm actually manipulates the time intervals between the DIO messages when the network remains unchanged.

Figs. 4, 5, 6, and 7 illustrate the differences in time interval with respect to the message index. We can notice that the largest difference is observed in Fig:5, where the longest time

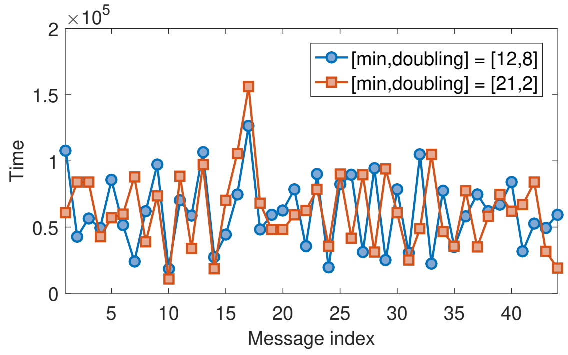

Figure 6: Performance comparison in terms of time taken for generation of DIO message for [min,doubling]=[12,8] and [21,2].

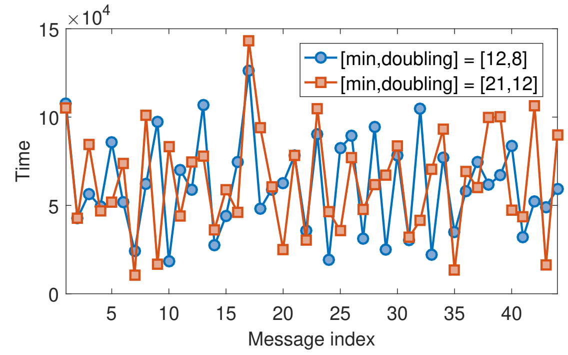

Figure 7: Performance comparison in terms of time taken for generation of DIO message for [min,doubling]=[12,8] and [21,12]

interval is experienced when $I_{\max}$ reaches maximum value for the pair [min, doubling]=[12,12] compared to the $I_{\max}$ for [min, doubling]=[12,8]. This difference is inline with the Fig: 2, where [min, doubling]= [12,12] value achieves the longest time while [12,8] results in third longest.

Furthermore, the comparison of time-intervals with the value pair [min, doubling] = [12,8] in Fig: 4,6, and 7 did not result in much difference even though the parameter values have a significant differences between them. This is because redundancy constant k-value is low, and is not sufficient to achieve such a large $I_{\max}$ value. In these three cases, the simulation could never achieve the calculated threshold of $I_{\max}$, as the counter expired and subsequently the messages are suppressed. Hence to achieve a much higher $I_{\max}$ value, the k value should also be increased.

Therefore, it can be concluded that the optimal values of these parameters vary depending on the network objective and underlying requirements. In case of an UAV root node with higher mobility, the optimal value for this factors depend on the data generation rate of network nodes. This determines how often the DIO messages are required, how often there is a possibility of a network inconsistency, understanding the behaviour and purpose of the sensors etc.

5 CONCLUSION

In this article, we have investigated the performance of RPL routing protocol in low-power lossy IoT networks assisted with an UAV with several constraints. RPL considers constraints and metrics defined using the objective function. The three primary control messages – DIO, DAO, and DIS and their rate of generation determines the power consumption. Therefore, in order to suppress the generation of DIO control message, trickle Timer algorithm uses DIO-INTERVAL-MINIMUM and DIO-INTERVAL-DOUBLING. We have performed an analysis to understand how these parameter affect the energy consumption. Finally, we have shown that optimal values for these parameters according to the network requirement can further optimise the working of the protocol for the data collection in IoT network.

REFERENCES

[1] Hazrat Ali. 2012. A performance evaluation of rpl in contiki.
[2] Cosmin Cobarzan, Julien Montavont, and Thomas Noel. 2014. Analysis and performance evaluation of RPL under mobility. In *2014 IEEE symposium on computers and communications (ISCC)*. IEEE, 1–6.
[3] Badis Djamaa and Mark Richardson. 2015. Optimizing the trickle algorithm. *IEEE Communications Letters* 19, 5 (2015), 819–822.
[4] Olfa Gaddour and Anis Koubâa. 2012. RPL in a nutshell: A survey. *Computer Networks* 56, 14 (2012), 3163–3178.
[5] Philip Levis, Thomas Clausen, Jonathan Hui, Omprakash Gnawali, and J Ko. 2011. The trickle algorithm. *Internet Engineering Task Force, RFC6206* (2011).
[6] José VV Sobral, Joel JPC Rodrigues, Ricardo AL Rabêlo, Jalal Al-Muhtadi, and Valery Korotaev. 2019. Routing Protocols for Low Power and Lossy Networks in Internet of Things Applications. *Sensors* 19, 9 (2019), 2144.
[7] Joydeep Tripathi, Jaudelice Cavalcante de Oliveira, and Jean-Philippe Vasseur. 2010. A performance evaluation study of rpl: Routing protocol for low power and lossy networks. In *2010 44th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 1–6.
[8] Tsvetko Tsvetkov and Alexander Klein. 2011. RPL: IPv6 routing protocol for low power and lossy networks. *Network* 59 (2011), 59–66.
[9] J Vasseur, Navneet Agarwal, Jonathan Hui, Zach Shelby, Paul Bertrand, and Cedric Chauvenet. 2011. RPL: The IP routing protocol designed for low power and lossy networks. *Internet Protocol for Smart Objects (IPSO) Alliance* 36 (2011).
[10] Tim Winter, Pascal Thubert, Anders Brandt, Jonathan W Hui, Richard Kelsey, Philip Levis, Kris Pister, Rene Struik, Jean-Philippe Vasseur, and Roger K Alexander. 2012. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. *rfc* 6550 (2012), 1–157.