Trickle-F: fair broadcast suppression to improve energy-efficient route formation with the RPL routing protocol

Carlo Vallati, Enzo Mingozzi
Dipartimento di Ingegneria dell'Informazione
University of Pisa, Italy
Email: {carlo.vallati, e.mingozzi}@iet.unipi.it

***Abstract*—RPL (IPv6 Routing Protocol for Low Power and Lossy networks) is a routing protocol recently standardized by the IETF. RPL has been designed to operate in energy-constrained networks with thousands of nodes, and therefore it is one of the most promising candidate routing protocols for Advanced Metering Infrastructure (AMI) networks. In this paper a performance evaluation of RPL is presented. An extensive study of the protocol is carried out with particular focus on Trickle, the algorithm adopted to control routing update distribution across the network. The performance of the protocol is analyzed considering different Trickle parameters in order to capture their impact on route formation and node power consumption. Results highlight that the non-deterministic nature of Trickle can lead to sub-optimal route formation especially when high message suppression is performed. In order to mitigate this issue, an enhanced version of the protocol, namely Trickle-F, is proposed in order to guarantee fair broadcast suppression. Trickle-F is demonstrated to be effective in obtaining more efficient routes with the same power consumption of the original version.**

***Keywords*-Low power lossy networks; RPL; Trickle; Advanced Metering Infrastructure; Wireless sensor networks.**

I. Introduction

The last years have seen an exponential growth of interest in the Internet of Things. Its vision foresees billions of devices connecting to the Internet in extremely heterogeneous contexts like building automation, city surveillance or health monitoring. A particularly challenging scenario is that of Advanced Metering Infrastructures (AMIs) [1], which are networked systems enabling measurement, configuration and control of electric, gas and water distribution systems. AMI networks are composed of a very large number of devices, including meters, distribution system elements, and home devices, which are distributed across a wide area comprising both urban and rural environments. Such devices communicate with each other using a variety of wired and wireless technologies, and are usually involved also in packet forwarding, thus forming a multi-hop network. A robust and scalable routing protocol is therefore needed to cope with the thousands of devices which can be in close proximity of each other in a typical AMI deployment (e.g., apartment buildings in urban centers). In addition, AMI deployments, e.g., for water and gas distribution, can also be energy-constrained, i.e., metering devices have no access to energy sources and therefore are battery-operated or energy-harvesting. In such a case, energy consumption is a critical factor to cope with the long expected lifetime of an AMI system, and routing mechanisms need to take it into account in order to minimize energy use and prolong system lifetime.

In order to prevent fragmentation in the market, a Working Group in the Internet Engineering Task Force (IETF) has been created with the goal of defining a "Routing Protocol for Low Power and Lossy Networks". The result of this standardization effort is the *RPL* routing protocol [2]. Because RPL was designed to operate in energy-constrained networks with thousands of nodes, its applicability in the deployment of AMI networks has been recently recognized [3].

RPL takes energy consumption into account by defining a number of energy-aware routing metrics. Moreover, it includes built-in energy-saving mechanisms in the route processing, like the *Trickle* algorithm [4] adopted to distribute route information across the network. *Trickle* aims in particular at minimizing the amount of route updates broadcast in the network while keeping low route convergence times. Trickle performance has been studied theoretically [5]. However, in real deployments link quality variations may cause message losses, which lead to long convergence periods or suboptimal routes [6].

In this work we carry out an extensive performance evaluation of RPL by means of simulation. Different Trickle parameters are considered in order to analyze their influence on the route formation. Our goal is to draw a set of guidelines to master the crucial trade-off characterizing low power and lossy networks: *route quality* vs. *energy consumption*. Simulation results show that the non-deterministic broadcast of route update messages driven by Trickle may bring to suboptimal route formation when high message suppression is performed. Originally designed as a gossiping algorithm that regulates the transmission of the same piece of information rapidly through a network, Trickle is not well suited for routing: the non-deterministic message suppression performed by Trickle can cause some nodes to remain silent for long periods resulting in some routes to remain undiscovered. In order to overcome this issue, an enhanced version of the original Trickle algorithm,

978-3-901882-56-2 ©2013 IFIP

named *Trickle-F*, is proposed in order to guarantee a fair broadcast suppression of route updates while facilitating the discovery of all available routes. By means of simulations, we demonstrate that Trickle-F is effective in guaranteeing better network routes with the same number of messages of the original version. Summarizing, our contribution is twofold: (i) a performance evaluation of RPL with respect to Trickle parameters which leads to a set of guidelines to optimize routes and power consumption; and (ii) a revisited version of Trickle that guarantees better routes with the same energy consumption.

The rest of the paper is organized as follows. Section II provides an overview of RPL and Trickle. The results of the performance evaluation of RPL are shown in Section III, while in Section IV Trickle-F is illustrated and its performance is assessed. Section V reports on related work, and finally some conclusions are drawn in Section VI.

II. RPL

RPL is a routing protocol specifically tailored for lossy environments and battery powered devices. The protocol takes into account the unreliable nature of the communications and the limited available power of the devices by minimizing the memory requirements and the complexity of the routing functionalities and reducing the signaling overhead. RPL operates at the IP layer and can be used across multiple type of link layers. In LLNs traffic is mainly directed to one *root* node acting as border router for data collection. On the contrary, traffic originated from the root node is sporadic, while node-to-node communication is rare. RPL design reflects this traffic pattern: RPL is a mixed proactive-reactive algorithm which supports pro-actively nodes-to-root traffic through the periodic emission of control messages to form upward routes, while it handles root-to-nodes traffic through reactive on-demand route formation. Node-to-node traffic is not directly supported and can be delivered only indirectly: the source node sends the message to the root which reroutes the message downward to the destination.

RPL adopts a distance vector routing algorithm which builds a logical topology on top of the physical network. In particular, the topology is a *Destination Oriented Directed Acyclic Graph*, *DODAG* for short. The root node of the DODAG emits first DODAG *Information Object* messages (hereafter *DIOs* for short). Non-root nodes listen for DIOs and use the included information to join one DODAG. As a node joins a DODAG, it starts advertising its presence through the emission of DIO messages. Each DIO message specifies the rank of the sender, which is a scalar measure of the distance of that node from the root. The rank must monotonically decrease on each path identified by the DODAG towards its root. Such property is used to form the logical topology and to avoid loops.

As DIO messages are received from the neighbors, each node updates its view of the topology. In particular, a set of neighbors with lower rank is selected to form a *parent set* which is used for data forwarding. Among them, a *preferred parent* is selected to forward traffic towards the root. Figure 1 shows an example of one DODAG. Each node selects a parent set (dashed and solid lines); among them a preferred parent is selected for upward traffic forwarding (solid line).

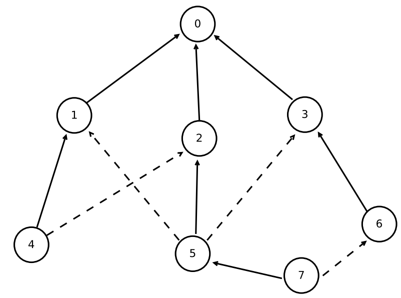

Figure 1. DODAG example

The rank is evaluated according to an *Objective Function* (*OF* for short), which defines how the rank is computed and updated. Although the rank is not meant as a path cost, it can be derived from path metrics dependent on the distance from the root, e.g., link quality, packet delay or number of hops. Several OFs have been defined, such as the Basic OF [7], and the *Minimum Rank with Hysteresis Objective Function* (*MRHOF*) [8], which aims at reducing route flaps caused by small metric fluctuations.

Each root node marks its DIOs with an unique DODAG ID. Since multiple roots are allowed in the same DODAG instance, a node potentially receives DIOs from multiple roots with different DODAG IDs. RPL allows each node to join only one DODAG per RPL instance.

DIO messages also specify a version number, which is used to identify a specific DODAG instance over time. Root nodes periodically increase this number in order to reset all the routing information in the network and re-create the DODAG from scratch. When a node receives a DIO with a new DODAG version number, it resets all the routing information and starts over the preferred parent selection procedure. This practice called *global repair* is defined to repair broken links and remove discontinued parents. Nevertheless, when a link or a node becomes disconnected, the children remain out of service until the the global reset. Since this time can be in the order of minutes or hours, a *local repair* procedure is also defined to fix unavailability of local routes in a short time.

Each node forwards upward traffic to its preferred parent. In order to support also downward traffic, nodes generate on-demand a *Destination Advertisement Object* message (hereafter *DAO* for short) which propagates destination information upward in the DODAG. Intermediate nodes process and forward the message according to the mode of operation of the network: *non-storing mode* and *storing mode*. In non-storing mode, intermediate nodes have limited resources and can not store routing information. In this mode source routing is performed: the root calculates the path by means of the information received in the DAOs. Intermediate nodes forward the message to one of their neighbors according to the path specified by the sender in the packet header. In storing mode, instead, each intermediate node stores a routing table and the classical hop-by-hop routing is executed.

A. The Trickle algorithm

The *Trickle* [4] algorithm, originally designed for polite gossiping in multi-hop wireless networks, has been adopted by RPL to regulate DIO broadcast transmission so as to reduce nodes' energy consumption.

The rationale behind Trickle is as follows. DIO broadcast transmission is performed by each node on a periodic basis. However, a node can suppress the transmission in a given period if enough DIO messages have been already overheard in the recent past from its neighbors. Moreover, the period is dynamically adapted depending on the 'steadiness' of the information carried by received DIO messages: if the information is deemed "consistent" with the current route configuration, the period length is exponentially increased. Otherwise, if it is not consistent, the period is reset to the minimum value to boost spreading of the new information. In this manner, promptness is preserved while the number of DIO messages transmitted overall is highly reduced, thus resulting in much energy saved by each node.

More in detail, Trickle operates on each node as follows. Each transmission period is halved into two subsequent time intervals: the *listening* interval and the *transmitting* interval, respectively. At the beginning of the period, a DIO message transmission is scheduled at a random time t in the *transmitting* interval. Up to t, the node keeps track of received messages by incrementing a *redundancy counter* c each time a new DIO message is received. At time t, if c is below a *redundancy threshold* k, the node actually transmits the DIO message, otherwise the transmission is suppressed. Finally, let τ represent the length of the current transmission period. At the end of the period, if only consistent messages were received, τ is doubled, i.e., the length of the next period is doubled with respect to the current one, until a maximum value I_{max} is reached. Otherwise, if an inconsistent message was received, τ is reset to a minimum value I_{min}. In any case, at the end of the interval, c is reset. Fig. 2 illustrates an example of Trickle operation with DIO transmission suppression in the case of four nodes and $k = 2$.

RPL specifies the conditions determining inconsistent events which cause the reset of the Trickle timer. In particular, a reset is mandatory when the DODAG version number of a DIO message is newer than the current one, i.e., when a global repair procedure has been triggered, or when a loop in data forwarding is detected. A reset in reaction to other events, like, e.g., a change in the parent set or a new

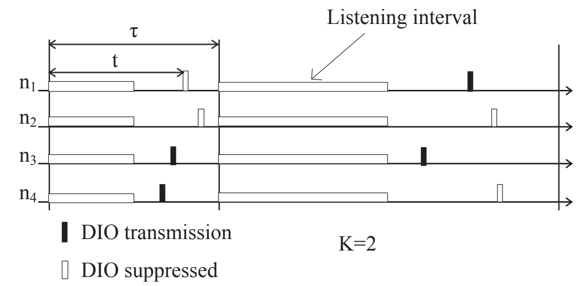

Figure 2. Example of Trickle operation with four nodes, $k = 2$.

value of the rank, is not mandatory and left implementation dependent [2]. In order to highlight how Trickle influences routing performance, our RPL implementation does not consider such events as inconsistent. The study of the impact of different reset policies is left for future work.

III. Performance evaluation

In this section we illustrate the results of the performance evaluation. Simulations are carried out by means of Omnet++ 4.2[1], a well known discrete event simulator. The *RPL* algorithm along with the *6lowpan* encapsulation and header compression mechanisms as defined in [9] have been implemented into the MiXiM-INET integrated framework. Two different topologies are considered: a regular topology, where nodes are placed forming a grid with a fixed distance in-between, and a random topology, where nodes are placed randomly following a uniform distribution inside a playground. All the nodes communicate on the same channel through a wireless transceiver equipped with the CC2420 chip[2]. The IEEE 802.15.4 MAC layer is configured in non-beacon mode with 64-bits addressing. IPv6 packets are encapsulated into IEEE 802.15.4 frames using 6LoWPAN [9] with HC2 compression.

In order to guarantee route stability, MRHOF is adopted as OF. The *expected transmission count* (ETX for short) [10] is selected as link cost metric. RPL unreliability in measuring the quality of links is highlighted in [11]: the absence of periodic probing traffic makes difficult to evaluate accurately the link costs which are measured only when data traffic is sent. In order to focus only on the evaluation of the routing algorithm, in our simulations we assume that the cost of each link is evaluated off-line and provided to the routing algorithm. For each topology, one single DODAG root node is selected.

The trade-off between route efficiency and node energy consumption is evaluated through the following metrics. The node *energy consumption* is derived from the transmission power consumption P_T, measured through the model proposed in [12]:

$$P_T = P_{T0} + P_{tx} \cdot \eta \tag{1}$$

[1]Omnet++, Discrete Event Simulation System, http://www.omnetpp.org

[2]CC2420, Single-Chip 2.4 GHz IEEE 802.15.4 Compliant RF Transceiver, http://www.ti.com/product/cc2420

Table IV
SYSTEM AND SIMULATION MODEL PARAMETERS

Name	Value
Transmission Power	1.0 mW
Receiver sensitivity	-94 dBm
Channel Frequency	2.4 GHz
Channel Model	Path Loss model
Path loss exponent	5 (indoor)
Pt0	26.5mW
η	0.0375

Table V
RPL AND TRICKLE PARAMETERS

Name	Range
I_{min}	2^3, 2^5, 2^7, 2^9, 2^{11} ms
I_{max}	$I_{min} \cdot 20$
T_{reset}	1035, 100663 s
k	1 - 15
Parent set size	1, 3

where P_{T0} is the constant energy consumption due to Digital Signal Processing (DSP) and front-end circuits during transmission, η is the drain efficiency and P_{tx} is the transmission power. The quality of the route towards the root node is measured through the *path stretch* defined as the difference between the route cost and the cost of the shortest path. In order to measure the quality of routing network-wide we define the *network stretch* as the ratio between the number of nodes with a patch stretch greater than 1 and the total number of nodes. The transient period between two DODAG resets is simulated in each run. Metrics are measured when the first DODAG is formed (all the nodes have joined the DODAG), and at the end of simulation (right before the next DODAG reset). Different values of simulation duration are considered in order to simulate different periods between two DODAG resets, T_{reset}.

Table IV summarizes the system and simulation parameters, Table V shows the RPL and Trickle parameters varied in our simulations.

A. Grid topology

In this first set of simulations a grid topology is considered. Two grids of different size are simulated: a 20x20 (400 nodes) grid and a 10x10 grid (100 nodes). In each topology, nodes are placed 10 meters far from each other. Since the results for the 10x10 grid confirmed the conclusions drawn for the 20x20 grid, for the sake of brevity we present only the results for the latter scenario. In order to obtain statistically sound results, 200 independent replications with different seeds are run for each scenario. The average value of each metric with its 5% confidence interval is reported.

Figures 3 and 4 illustrate the performance of the first DODAG formation. Figure 3 shows the formation time, i.e. the time needed for all the nodes to receive at least one DIO message. As can be seen, the lower I_{min} is, the sooner the first DODAG is formed and the sooner all the nodes can send upward traffic, i.e., short communication intervals

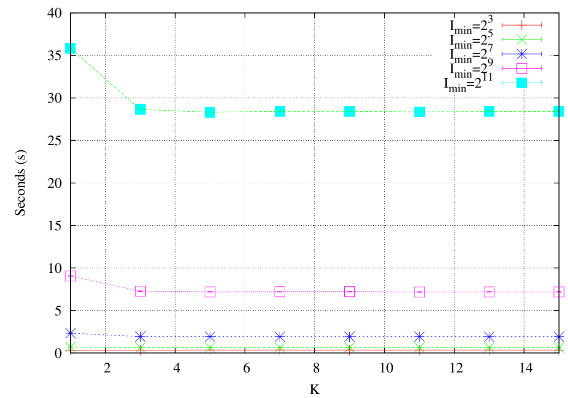

Figure 3. Average formation time first DODAG, grid topology

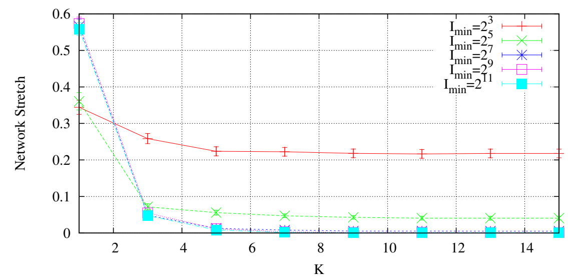

Figure 4. Average network stretch, first DODAG, grid topology

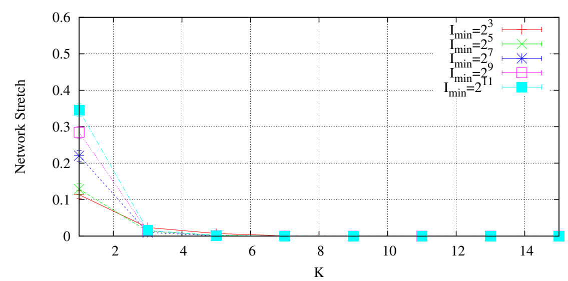

Figure 5. Average network stretch, $T_{reset} = 1035s$, grid topology

facilitate a quick formation of the first DODAG, because more messages are broadcast in the same amount of time. Formation time is only slightly influenced by k, considering the regular topology of the grid the message suppression does not delay significantly the discover of at least one route by each node. Figure 4 shows the quality of the routes of the first DODAG. As expected, the quality is mainly influenced by I_{min}: higher I_{min} values result in longer formation time, however, with better routes. The quality of the routes depends also on the suppression coefficient k: the lower k is, the higher the network stretch is. As nodes suppress messages, less paths can be discovered during the first DODAG formation.

Figures 5 and 6 illustrate the average network stretch when T_{reset} is set to 1035s (17 min) and 100663s (1677 min), respectively. As for the first DODAG, the higher the

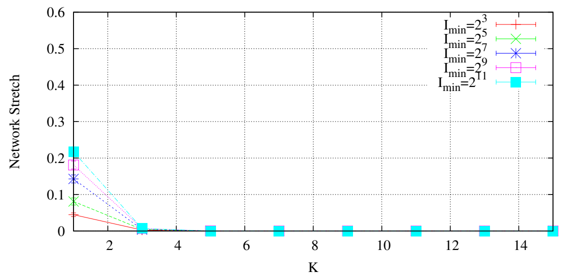

Figure 6. Average network stretch, $T_{reset} = 100663s$, grid topology

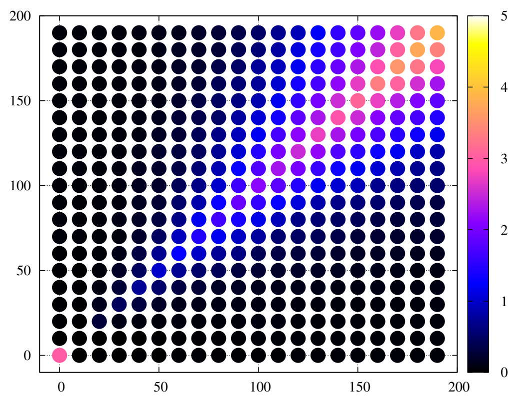

Figure 7. Average path stretch network map, $I_{min} = 2^{11}$, $K = 1$, $T_{reset} = 1035s$, grid topology

message suppression is, the higher the network stretch results. As expected, as more messages are broadcast through the network, new shorter paths are discovered, hence less nodes suffer from suboptimal routes. As I_{min} decreases the overall network route stretch reduces. This can be explained considering the fact that more messages are sent over the same period of time, which allows more routes to be discovered. These results show how Trickle can lead to suboptimal routes especially with low values of k, even with large T_{reset} values. This can be explained by looking at the spatial distribution of the path stretch. Figure 7 illustrates the average path stretch of each node when $I_{min} = 2^{11}$, $K = 1$ and $T_{reset} = 1035s$, the scenario in which this behaviour is more stressed. As can be seen several nodes do not discover the shortest path, some of them discover only routes with a high stretch. This can be explained considering the fact that they receive less messages then the others and hence the shortest path remains undiscovered.

Figure 8 illustrates the other side of the trade-off: the average energy consumption per node. As can be seen, the shorter the I_{min} is, the larger the number of sent messages is, hence resulting in a higher energy consumption per node. The energy consumption is strongly influenced by the value of k: the lower it is, the lower the energy consumption is. As k increases from 1 to 13, the energy consumption of each node increases since less transmissions are suppressed. With k equal to 13, the energy consumption reaches its maximum

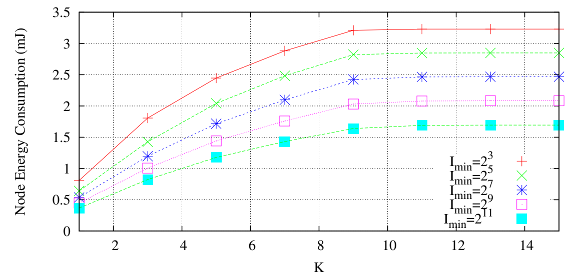

Figure 8. Average energy consumption per node (mJ), $T_{reset} = 1035s$, grid topology

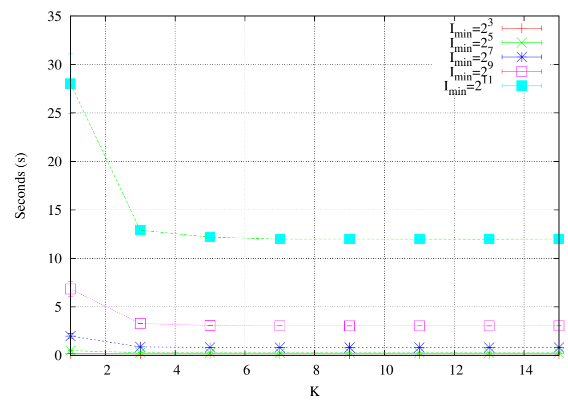

Figure 9. Average formation time first DODAG, uniform topology

value, i.e., no suppressions are performed and packets are always forwarded regardless of the number of consistent transmissions received.

B. Random topology

In this second set of simulations an irregular random scenario is considered. Following a uniform distribution, 100 nodes are placed in a rectangular playground with 100m side. The obtained topology is verified to be connected, i.e., every node has at least one neighbor and one path to reach the root node. Links with ETX greater than 5 are excluded from routing and shortest path evaluation. In order to compensate for the higher randomness of this topology, 1000 independent replications with different seeds are run for each scenario.

Figure 9 shows the first DODAG formation time. The same conclusions drawn for the grid topology still hold. It is important to highlight how the random topology benefits more than the regular one from using higher k values, i.e., the formation time decreases more rapidly as the suppression threshold is increased. This can be explained by considering the quality of the links in this topology: rather than a grid, a random topology is characterized by a high heterogeneity of the quality of its links. The high packet loss of bad links delays the discovery of certain routes, i.e., lossy links require several messages to be sent before being discovered. For this reason, isolated nodes connected to the rest of the

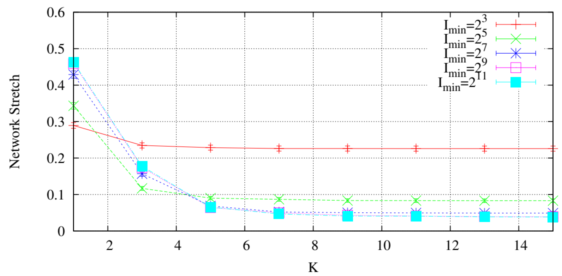

Figure 10. Average network stretch, first DODAG, uniform topology

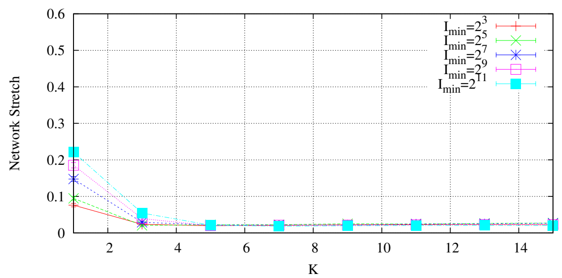

Figure 11. Average network stretch, $T_{reset} = 1035s$, uniform topology

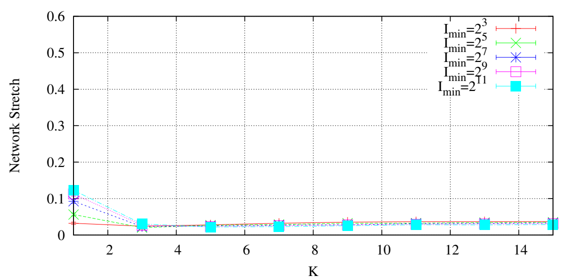

Figure 12. Average network stretch, $T_{reset} = 100663s$, uniform topology

network only through links with high ETX require more time to join the DODAG. As nodes broadcast more messages, the probability for those nodes to receive a DIO message increases, thus reducing the time for the DODAG formation.

Figure 10 shows the network stretch at the time of the first DODAG. Similar to the grid case, when low k values are adopted, half of the nodes are characterized by suboptimal routes. Although also in this scenario the network stretch depends on k and I_{min}, the presence of links with high ETX values increases the amount of messages necessary to improve the quality of the routes, i.e., higher values of k reduce less dramatically the number of nodes with suboptimal routes in comparison with the results obtained with the grid topology.

Figures 11 and 12 illustrate the network stretch at the end of the simulation with two different T_{reset}, 1035s and 100663s respectively. As can be seen, the same behavior displayed in the grid scenario is shown. However, it is important to notice that the shortest path can not be reached even with high k values. This can be explained with the presence in the shortest paths of links with very high ETX values, which are likely included since they can cover large distances and allow the sink node to be reached with few hops. The discovery of these routes, however, requires a long time which is beyond the T_{reset} period durations considered in our simulations.

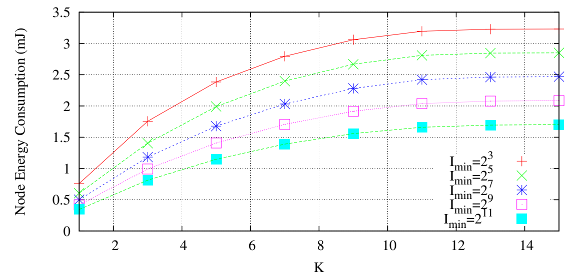

Figure 13. Average energy consumption per node (mJ), $T_{reset} = 1035s$, uniform topology

Finally, Figure 13 shows the average energy consumption for each node. Since this metric is strictly dependent on the number of packets transmitted, the obtained results are the same as in the grid case. A close comparison with the results of the grid scenario highlights a slightly higher energy consumption for low k values. This can be explained by considering the lower density of nodes which causes a higher number of transmissions also with low values of the suppression threshold.

C. *Trickle setting guidelines*

An analysis of the obtained results at large can provide a set of guidelines to tune Trickle parameters. Low I_{min} values result in the creation of the first DODAG within a short space of time, however its routes are suboptimal and need extra time to be refined. This behavior highlights a trade off *time* vs *route quality* for the formation of the first DODAG, suggesting that the I_{min} value should be tuned according to the T_{reset}: if RPL is characterized by frequent DODAG resets (low T_{reset} values), the priority is to form the DODAG as soon as possible regardless of its quality. On the contrary, if higher reset periods are adopted, larger values can be selected. The redundancy threshold k, instead, can be used to regulate the trade-off between energy consumption and route quality: low k values reduce the energy consumption of each node, resulting however in suboptimal routes.

IV. TRICKLE-F: FAIR BROADCAST SUPPRESSION

Simulation results presented in Sect. III suggest that suppressing DIO message broadcast by means of Trickle may lead to sub-optimal route formation. The more Trickle parameters are configured to suppress a larger number of messages in each neighborhood, the more this phenomenon is noticeable.

We explain this behavior as follows. Trickle was originally designed as a gossiping algorithm: its original goal was to spread the same piece of information across a network rapidly with a minimum number of messages. Routing information updates, as those carried by DIO messages, are instead strictly dependent on the source of the message: suppressing one transmission or another is not always equivalent, since the two suppressed messages carry different information. Should some node be not allowed to send any message for a long time, some routes may remain undiscovered and therefore unused for such a time even though they are better than those currently active in the DODAG. The original Trickle algorithm provides each node with equal average broadcast transmission probability in the long run. However, we claim that for routing purposes it is important that every node is given the opportunity to share its routing information in the shortest possible time scale, so as to allow the quick discovering of all available routes, and then choose the best ones according to the established routing metrics.

Based on previous observations, we propose a modified version of the Trickle algorithm, i.e., *Trickle-F*, which aims at guaranteeing a fair short-term broadcast suppression among nodes in a neighborhood in order to facilitate the rapid discovery of all available paths. The rationale behind Trickle-F is to prioritize each node strictly depending on the number of consecutive suppressions: the longer the time spent by a node without transmitting, the higher its transmission priority in the next round. In order to achieve this, Trickle-F introduces a modification to the original algorithm in the computation of the start time and length of the next transmission period.

A pseudo-code of Trickle-F is presented in Algorithm 1. The modifications with respect to the original design are highlighted through "+" and "-" signs, to indicate a statement that has been added or removed, respectively. In order to provide broadcast fairness, each node keeps track of s, the number of continuous communication intervals in which a message transmission has been suppressed. At time t, if a DIO is transmitted, s is reset; otherwise, the counter is incremented. Each node gets a transmission priority proportional to the number of last consecutive suppressed transmissions. Priority dependence on s is enforced by modifying the length of the listening and transmitting intervals. Each transmission period τ is not halved as in the original algorithm: the listening and transmitting periods are set to a variable length which is proportional to the number of suppressed transmissions, $\frac{\tau}{2^s}$. At the beginning of each period, the transmission instant t is selected in a sub-period depending on s as follow: $[\frac{\tau}{2^{s+1}}, \frac{\tau}{2^s}]$. This ensures strict prioritization of the sub-periods according to s: the larger s is, the closer the sub-period is. This modification guarantees that nodes that have waited longer get higher transmission probability, while nodes that have been suppressing the same number of transmissions will have the same transmission probability. It is important to highlight that each node still has a listening sub-period equal to $\frac{\tau}{2^{s+1}}$: this allows to overcome the *short-listen* problem as in the original design [13].

Algorithm 1 Trickle-F

```
function INITIALIZATION()
        τ ← I_min
    +   s ← 0
function INTERVALBEGINS()
        c ← 0
    -   t ← random(τ/2, τ)
    +   t ← random(τ/2^(s+1), τ/2^s)
function CONSISTENTTRANSMISSIONRECEIVED()
        c ← c + 1
function TIMEREXPIRES()
    if k ≥ c then
            Transmit DIO
        +   s ← 0
    else
        +   s ← s + 1
    end if
function INTERVALENDS()
        c ← 0
    if InconsistentTransmissionReceived then
            I ← I_min
        +   s ← 0
    else
            I ← I × 2
        if I_max ≤ I then
                I ← I_max
        end if
    end if
```

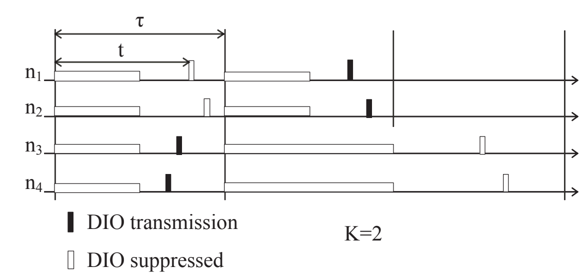

Figure 14. Example of Trickle-F operation with four nodes, $k = 2$.

Fig. 14 illustrates an example of Trickle-F operation in the same scenario as that in Fig. 2 in the case of four nodes and $k = 2$. This time, nodes 1 and 2 schedule DIO transmission in a shorter next transmission period (as a matter of fact, during the listening interval of nodes 3 and 4) and therefore get priority over nodes 3 and 4, which transmitted in the previous period.

A. Performance evaluation

In order to evaluate our proposal, we run the same set of simulations in both regular and random topologies. Figures 15 and 16 illustrate the network stretch in the grid

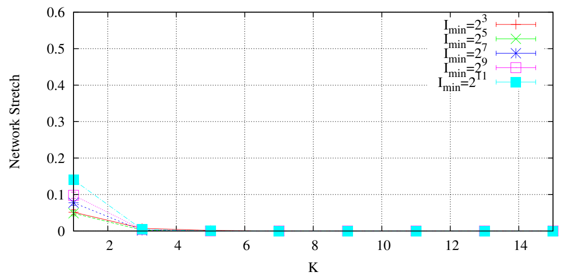

Figure 15. Average network stretch Trickle-F, $T_{reset} = 1035s$, grid topology

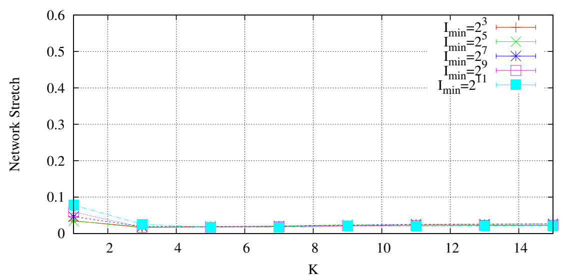

Figure 16. Average network stretch Trickle-F, $T_{reset} = 1035s$, uniform topology

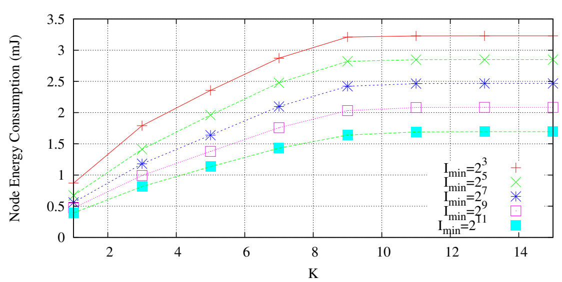

Figure 17. Average energy consumption per node (mJ) Trickle-F, $T_{reset} = 1035s$, grid topology

and random scenarios, respectively. As can be seen the modifications succeed into reducing the number of nodes with suboptimal routes, i.e., better routes are discovered by the routing algorithm for the same value of k. As shown in Figures 17 and 18, the same energy consumption of the original Trickle is guaranteed. This advantage can be exploited to obtain better routes with the same energy consumption, or to obtain the same quality of routes with lower energy consumption.

Better routes are achieved through the spatial fairness among the nodes. Figure 19 shows the average number of suppressed transmissions for each node, on the left with the original Trickle, on the right with Trickle-F. Apart from the nodes at the border which suppress less transmissions due to their reduced neighborhood size, Trickle-F succeeds into guaranteeing a more fairness in the distribution of the suppressed transmissions all over the network.

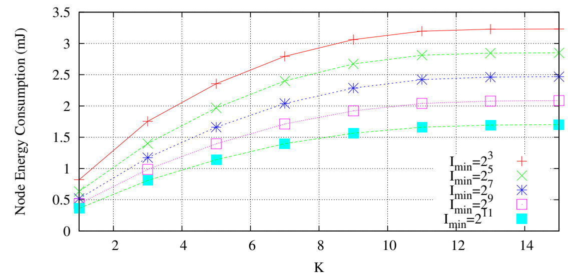

Figure 18. Average energy consumption per node (mJ) Trickle-F, $T_{reset} = 1035s$, uniform topology

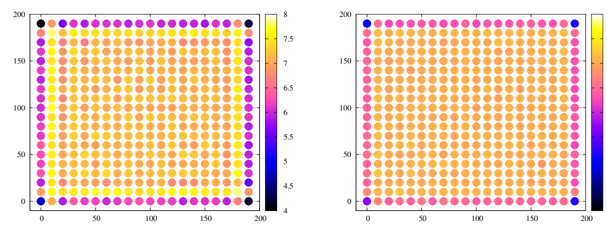

Figure 19. Average suppressed DIO messages per node trickle normal vs Trickle-F, $I = 2^{11}$, $K = 1$, grid topology

V. Related Work

Recently the interest in low power and lossy network increased, as RPL implementations for operating systems like Contiki[3] and Tiny OS[4] testify. As the interest in RPL has grown, a few studies have been carried on with the goal of assessing its performance. Almost all of them, evaluate the performance of the routing protocol at the steady state, without considering the transitory phase between two DODAG resets. Each work usually focuses on a specific case of use. A performance evaluation specifically tailored for smart metering systems is presented in [14], [15], [16]. However, RPL is evaluated only considering high level metrics such as end-to-end traffic-related metrics and not sensor specific onse such as power consumption.

The work in [17] evaluates RPL in comparison with a different routing protocol, LOAD. The latter is derived from AODV and supports different types of traffic flows. The influence of ETX link estimation on RPL stability is evaluated in [18]. To the best of our knowledge, the work in [18] is the first one highlighting the possibility that the algorithm can have long convergence times, especially when certain settings are adopted. However, its focus is only on link quality estimation. A performance evaluation of RPL is presented in [19], which specifically considers the impact of the Trickle timer on the performance. However, Trickle is compared with periodic message broadcasting without considering its parameters. Both [20] and [11] consider the performance of the DODAG construction in their analysis. However, Trickle parameters are kept fixed in the evaluation.

[3] http://www.contiki-os.org/
[4] http://www.tinyos.net/

Differently from the other works proposed in literature, we evaluated how Trickle dynamics influence the DODAG formation in RPL, and how this is influenced by its settings. To the best of our knowledge, this is the first work considering different parameters and studying their influence in the overall performance of the system.

VI. Conclusions

In this work we presented a performance evaluation of RPL with particular focus on Trickle and its parameters. Simulation results showed how the routing behavior is affected by Trickle settings. A set of guidelines was derived from simulation results to optimize network formation time, route quality and energy consumption.

Simulation results highlighted how the transmission suppression policy can lead to suboptimal route formation. In order to overcome this issue, we proposed a set of modifications to the original algorithm in order to increase the spatial fairness in message transmission and facilitate the discovery of all the routes in the network. Simulation results demonstrated the validity of our proposal.

Acknowledgment

The authors would like to thank Federico Consoli for his contribution to the design and the realization of the RPL simulator.

References

[1] D. Hart, "Using ami to realize the smart grid," in *Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century, 2008 IEEE*, 2008, pp. 1–2.

[2] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550, Mar. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6550.txt

[3] D. Popa, J. Jetcheva, N. Dejean, R. Salazar, J. Hui, and K. Monden, "Applicability Statement for the Routing Protocol for Low Power and Lossy Networks (RPL) in AMI Networks," Internet-Draft, Tech. Rep., 2012. [Online]. Available: http://tools.ietf.org/html/draft-ietf-roll-applicability-ami-06

[4] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko, "The Trickle Algorithm," RFC 6206 (Proposed Standard), Mar. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6206.txt

[5] M. Becker, K. Kuladinithi, and C. Grg, "Modelling and simulating the trickle algorithm." in *MONAMI 2011*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 97. Springer, 2011.

[6] T. Clausen, U. Herberg, and M. Philipp, "A critical evaluation of the ipv6 routing protocol for low power and lossy networks (rpl)," in *IEEE WiMob 2011*, oct. 2011, pp. 365 –372.

[7] P. Thubert, "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)," RFC 6552 (Proposed Standard), Mar. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6552.txt

[8] O. Gnawali and P. Levis, "The Minimum Rank with Hysteresis Objective Function," RFC 6719(Proposed Standard), Mar. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6719.txt

[9] N. Kushalnagar, G. Montenegro, D. E. Culler, and J. W. Hui, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks," Tech. Rep. 4944, Sep. 2007. [Online]. Available: http://www.rfc-editor.org/rfc/rfc4944.txt

[10] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wirel. Netw.*, vol. 11, no. 4, pp. 419–434, Jul. 2005.

[11] E. Ancillotti, R. Bruno, and M. Conti, "Rpl routing protocol in advanced metering infrastructures: An analysis of the unreliability problems," in *IEEE SustainIT 2012*. IEEE, 2012, pp. 1–10.

[12] Q. Wang, M. Hempstead, and W. Yang, "A realistic power consumption model for wireless sensor network devices," in *IEEE SECON 2006*, vol. 1, Sept., pp. 286–295.

[13] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: a self-regulating algorithm for code propagation and maintenance in wireless sensor networks," in *NSDI 2004*, ser. NSDI'04, Berkeley, CA, USA, 2004, pp. 2–2.

[14] N. Bressan, L. Bazzaco, N. Bui, P. Casari, L. Vangelista, and M. Zorzi, "The deployment of a smart monitoring system using wireless sensor and actuator networks," in *SmartGrid-Comm 2010*, oct. 2010, pp. 49 –54.

[15] D. Wang, Z. Tao, J. Zhang, and A. Abouzeid, "Rpl based routing for advanced metering infrastructure in smart grid," in *ICC 2010*, may 2010, pp. 1 –6.

[16] J. Tripathi, J. de Oliveira, and J. Vasseur, "Applicability study of rpl with local repair in smart grid substation networks," in *SmartGridComm 2010*, oct. 2010, pp. 262 –267.

[17] U. Herberg and T. Clausen, "A comparative performance study of the routing protocols load and rpl with bi-directional traffic in low-power and lossy networks (lln)," in *ACM PE-WASUN 2011*, ser. PE-WASUN '11. New York, NY, USA: ACM, 2011, pp. 73–80.

[18] S. Dawans, S. Duquennoy, and O. Bonaventure, "On link estimation in dense rpl deployments," in *IEEE SenseApp 2012*, Florida, USA, October 2012.

[19] T. Clausen and U. Herberg, "Multipoint-to-point and broadcast in rpl," in *NBiS 2010*, pp. 493 –498.

[20] O. Gaddour, A. Koubaa, S. Chaudhry, M. Tezeghdanti, and M. Abid, "Simulation and Performance Evaluation of DAG Construction with RPL," in *COMNET 2012*, April 2012.