Computer Communications 81 (2016) 61–71

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Cooperative interaction among multiple RPL instances in wireless sensor networks

Marc Barcelo*, Alejandro Correa, Jose Lopez Vicario, Antoni Morell

Telecommunications and Systems Engineering Department, Universitat Autònoma de Barcelona, Spain

ARTICLE INFO

Article history:
Received 21 January 2015
Revised 24 December 2015
Accepted 26 December 2015
Available online 31 December 2015

Keywords:
Wireless Sensor Networks
Routing
RPL
Multiple instances
Heterogeneous traffic

ABSTRACT

Advanced Wireless Sensor Networks (WSNs) applications may need to develop multiple tasks that involve sensing, processing and gathering data from different sensing units. This heterogeneous data may have multiple and sometimes opposite sets of requirements. In these scenarios, different networking strategies must be combined, and therefore traditional single-tree routing approaches are not efficient. On the contrary, the well-known RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks) protocol virtually splits the network into multiple RPL Instances, that transport each kind of data according to its particular objective function. However, this protocol does not define any mechanism to decide the nodes that must belong to each instance, and this decision has a strong impact in the network energy consumption and performance. With this in mind, in this paper we introduce C-RPL (Cooperative-RPL). This creates multiple instances following a cooperative strategy among nodes with different sensing tasks. As a result, the energy consumption, the complexity and the cost of the nodes is reduced compared to RPL, since they are active less time, perform fewer tasks and are equipped with less sensing hardware. In this paper we also propose a novel fairness analysis for networks with multiple instances, showing that C-RPL achieves a better tradeoff, in terms of performance and energy consumption, than RPL with non-cooperative instances.

1. Introduction

Wireless Sensor Networks (WSNs) are a set of autonomous, battery powered nodes connected by wireless links that gather information about the environment. These networks are being used in many scenarios thanks to its reduced cost. However, in many of these scenarios, such as a forest or a battlefield, the replacement of their batteries can be difficult or even impossible. Then, it is necessary to minimize their energy consumption in order to extend their lifetime and keep them operative as much as possible. It is well-known that the main source of energy consumption in WSNs is the radio transceiver. Therefore, practical energy efficient communication schemes are required in order to extend their lifetime in real-life scenarios.

Many advanced WSN applications need to develop multiple tasks. For instance, SHM (Structure Health Monitoring) [1] systems need to collect information coming from different sensing units, such as pressure, vibration or temperature. Moreover, they also need to send alarm messages in case of broken sections or systems failures. In addition, continuous messages are broadcast for external monitoring and calibration. All this heterogeneous traffic needs to be properly managed by the network [2]. For instance, latency is a critical requirement in event detection applications [3]. A minimum hop strategy is frequently adopted in these applications, since packets need to be decoded, processed and coded again in each hop. On the contrary, critical monitoring tasks may admit a certain delay in some cases, but they require a high reliability [4]. The performance of these applications strongly depends on the packet delivery ratio at the destination, and therefore only the most reliable links should be considered. On the other hand, ambient monitoring applications may not have strict delay or reliability constraints, but a low energy consumption becomes crucial because periodic packet transmissions are generally required [5].

The well-known RPL (Routing Protocol for Low-Power and Lossy Networks) protocol [6] is a reliable and efficient routing protocol that can be easily configured with personalized objective functions. Moreover, multiple objective functions can be considered in the same WSN by virtually splitting the network into multiple RPL Instances, which group nodes with common traffic requirements. Then, each instance can be individually configured to address the QoS (Quality of Service) requirements of a specific kind of traffic.

* Correspondenece to: Edifici Q, Campus de la UAB (Bellaterra), 08193 Cerdanyola del Vallès (Barcelona), Spain. Tel.: +34 93 586 8114.
E-mail address: Marc.Barcelo@uab.cat (M. Barcelo).

http://dx.doi.org/10.1016/j.comcom.2015.12.008
0140-3664/

Unfortunately, RPL does not define any mechanism to create RPL Instances according to the node distribution and network conditions, and hence they must be defined manually in advance. In order to divide the network in RPL Instances, the network designer could either adopt a low-consumption strategy (i.e. each node belongs only to the instance associated to its tasks) or a high-reliability strategy (i.e. any node may belong to any instance as long as it senses or forwards the kind of traffic associated to that instance). On one hand, the first strategy may reduce the network performance, due to the reduction of the node density of each instance. On the other hand, the second strategy does not consider that nodes associated to different tasks may have different duty cycles. As a result, this solution may not be energy efficient, since it does not prioritize the communication among nodes with the same duty cycle. Then, many nodes may have to extend their active time, and thus increasing their energy consumption.

In this paper, we propose a cooperative RPL-based strategy (C-RPL) to manage this tradeoff. This defines the nodes that belong to each RPL Instance, referred in C-RPL to as C-RPL Instances, following a cooperative strategy among instances. Taking into account the selfish nature of nodes, the coalitions among the instances are created according to a utility function that considers the tradeoff between the performance and the energy consumption associated to each coalition. From a game theoretical perspective, the solution of the cooperation problem among RPL Instances, such as the solution of the WSN cooperation problem in [7], is very similar to the well-known prisoner's dilemma game [8]. Briefly, this is a two person zero game that describes a situation where two players increase their utility if they both cooperate, but if a player decides not to cooperate while the other cooperates, its utility gain is even higher than cooperating. Therefore, players will never cooperate (i.e. Nash equilibrium of the prisoner's dilemma game). This game is suitable for studying complex interactions among players, such as the cooperation among RPL instances, since rational actions do not cause the Pareto optimality. In C-RPL, we avoid that instances do not collaborate using the sink node as a supervising entity.

On the other hand, since multiple performance criteria may be involved, it is important to distribute the network resources in a "fair" manner (i.e. considering the different requirements of all the traffics in the network). Although many definitions can be found in the literature to evaluate fairness [9], such as weighted fairness, max-min fairness or proportional fairness, to the best of the authors knowledge, not any of these definitions has been used before when different objective functions are considered in the same network. In this paper, we propose a metric to evaluate the overall network fairness in networks with multiple instances, which may have different objective functions.

The main contributions of this paper are as follows:

- We address the performance and energy efficiency issues that may appear in RPL in the presence of heterogeneous traffic. Then, we propose a novel approach (C-RPL) that coordinates the RPL Instances to form energy efficient coalitions according to their individual objective functions and the network conditions.
- We propose a mechanism to evaluate fairness in networks with multiple RPL Instances. This evaluates the distribution of the existent network resources to address the different, and sometimes contradictory, objective function of each instance.

The rest of the paper is organized as follows: Section 2 introduces previous work related to the management of heterogeneous traffic in WSNs. Section 3 describes the principles of the RPL protocol. Section 4 introduces C-RPL, explaining how instances evaluate the potential coalitions, and also the cooperation game among these instances. Section 5 proposes a metric to evaluate fairness in networks with multiple instances. Section 6 presents and analyzes the simulation results. Finally, Section 7 summarizes the paper and presents the main conclusions.

2. Related work

Multi-objective routing approaches, such as [10,11], consider multiple criteria simultaneously to handle different QoS requirements. These strategies find a tradeoff solution taking into account multiple objective functions at the same time. For instance, in [12] the authors propose a solution to increase the lifetime and throughput of the network, while reducing its latency. In [13], the aggregate energy consumption and delay are optimized using genetic algorithms. However, the requirements of different kinds of traffic may not be satisfied with multi-objective routing, since they are not addressed individually. Note that frequently, these requirements have contradictory relationships among them [14].

The QoS requirements of multiple kinds of traffic can be individually addressed through buffering or prioritized Medium Access Control (MAC) mechanisms [15]. At the network layer, it is also possible to provide QoS differentiation through multiple routing trees. Although this mechanism has been mainly used in the literature for load balancing [16] or to increase the network robustness in front of faulty links [17], it can also be used to efficiently manage the QoS requirements of heterogeneous traffic [18]. In fact, RPL divides the network in multiple trees, referred to as RPL Instances, to enable the network to manage traffics with different requirements. In [19], two RPL instances are defined to individually manage latency constrained and high priority traffic in Smart Grids. In particular, they construct each RPL Instance according to the minimum number of hops and the minimum expected number of transmissions (ETX), respectively. In [20], the authors differentiate between nodes for monitoring purposes, and nodes for high priority traffic (alarms). In order to manage these traffics, they create one instance that is only composed of nodes from the first group, and also a second instance that groups nodes from both groups. However, any of these strategies define how to dynamically select the nodes that belong to each instance, and therefore this decision must be taken in advance. Note that the best solution may strongly depend on the current network conditions.

3. Principles of RPL

Gradient-based routing is an energy efficient and reliable solution to construct convergecast trees. This technique uses control messages (also referred to as pilots or beacons) to evaluate the quality of the wireless links. The link quality estimator metric can be based on different criteria, such as reliability, number of hops, node battery level, energy consumption, and so on. The particular metric should be selected according to the application requirements. GBR (Gradient-Based Routing) is the classical gradient routing protocol, but there also exist other gradient-based implementations, such as CTP (Collection Tree Protocol) [21], RPL [6], and the hierarchical routing defined in ZigBee [22].

In this paper, we consider RPL, since it is specially designed for Low-Power Lossy Wireless Networks (LLNs), such as WSNs. This has been standardized by the IETF ROLL working group. RPL constructs convergecast trees, referred in RPL to as DODAGs (Destination Oriented Acyclic Graphs), rooted at the sink node. In the presence of multiple kinds of traffic, RPL may create multiple instances to serve different and potentially antagonistic objective functions [19]. Each RPL Instance uses a unique objective function, but it may include one or multiple DODAGs constructed with this objective function. For the sake of simplicity, in this paper we assume that each RPL Instance includes only one DODAG. According to the particular objective function of each DODAG, the nodes compute their relative distance to the sink, referred

to as rank. This objective function must be defined according to the QoS requirements of the particular application (i.e., energy consumption, reliability, delay and so on). The typical objective function in RPL is the minimum number of expected transmissions (ETX). Basically, this metric indicates how many times a message must be transmitted in average to reach its final destination.

RPL defines three control message types: DODAG Information Object (DIO), DODAG Information Solicitation (DIS) and Destination Advertisement Object (DAO). Nodes broadcast DIO packets to propagate their rank in order to construct and maintain the DODAG. A DIO packet includes information that allows a node to discover a RPL Instance, learn its configuration parameters, select a DODAG parent set, and maintain the DODAG. The most relevant fields in a DIO packet are the 128-bit *DODAGID*, that uniquely identifies the DODAG, the 8-bit *RPLInstanceID*, that defines the instance of the DODAG, and the 16-bit *Rank*, that specifies the rank of the node. The rate of DIOs is managed by a trickle timer, that increases or decreases it dynamically. DIS packets are sent in unicast and may be used to solicit DIOs from a neighbor node, for instance to discover nearby DODAGs. When a node receives a DIS packet from another node, it must unicast a DIO to this node. Finally, DAO packets are sent in unicast upwards along the DODAG to the selected parent (storing mode), or to the DODAG root (non-storing mode). They are used to collect information about the network topology. For a detailed explanation of this protocol, the interested reader is referred to [6].

In this article, we assume that RPL is built on top of a simple low-power MAC strategy to better observe the impact of the proposed routing approach. This uses a slotted listening duty cycling scheme, and hence defines periodic sleep/active time slots according to the traffic requirements, such as in S-MAC (SensorMAC) [23]. In the presence of multiple instances, this assigns a personalized sleep/active pattern to each instance according to its particular QoS requirements. On one hand, the active time of two different instances is not overlapped to avoid interferences among their nodes. On the other hand, a TDMA-based scheme is applied to avoid interferences among the nodes that belong to the same instance. Note that a node that belongs to multiple instances must be active during the active time periods assigned to each of the instances that it belongs to. Therefore, its total active time is the sum of the active assigned to each of these instances.

4. C-RPL: Cooperative-RPL

In general terms, C-RPL coordinates the collaboration among RPL Instances, considering their selfish nature. Each instance aims to maximize its performance with the minimum energy consumption. In this section, we first compare RPL with C-RPL. Then, we explain how instances evaluate their utility related to each possible coalition. Finally, we explain the cooperation game among RPL Instances that is used to define the C-RPL Instances.

4.1. RPL with multiple instances vs. C-RPL

In RPL, instances operate independently from each other, and therefore their construction is also independent. The nodes that may belong to each instance need to be defined in advance, reducing the flexibility of the routing process. A particular RPL Instance may be either composed of: (i) the nodes with tasks associated with this instance, or (ii) all the nodes in the network. Assuming a slotted listening duty cycling scheme, in the first case nodes only need to be active during the active time assigned to their respective instance or instances. In the second case, any node may belong to any instance as long as it senses or forwards traffic associated to that instance. Therefore, nodes need to be active during the time assigned to its traffic, and also during the active time assigned to the rest of instances in which they participate. Clearly, the first strategy may reduce the network performance due to the reduction of the node density of each instance. On the other hand, the second strategy may waste energy resources because of including nodes with different duty cycle in the same DODAG. As a result, many nodes have to extend their active time, and thus increasing the average energy consumption of the network.

In LPL (Low Power Listening) MAC mechanisms, such as B-MAC [24], X-MAC [25] or ContikiMAC [26], nodes periodically wake up and check for activity on the channel, and stay awake only if they need to receive or transmit packets. These protocols use adaptive preamble sampling before transmitting a packet, in which the preamble length and the checking period need to be configured according to the energy consumption and latency requirements [24]. Although in this case the active time does not depend on the number of instances that each node belongs to, a similar tradeoff arises due to the preamble sampling mechanism. Since the network instances are independent from each other, they may have their own configuration parameters (e.g., sampling period, frequency channel), which must be known by any node that belongs to the instance. Then, the nodes that belong to multiple instances may need to increase their sampling frequency. Therefore, in order to avoid that nodes unnecessarily waste their energy resources, it is important to consider the particular network conditions and QoS requirements to efficiently define the instances that each node belongs to. Nevertheless, for the sake of clarity in this article we assume a simpler MAC layer.

In C-RPL, the nodes that belong to each instance do not need to be predefined, and therefore C-RPL Instances can be constructed dynamically according to the objective function of each instance, the location of nodes and the particular network conditions. In general, C-RPL defines the nodes of each C-RPL Instance following a cooperative approach among the nodes that develop the same task in the network. The objective of each instance is to maximize its utility, taking into account the possible coalitions that can be formed with the rest of instances in the network. We define a coalition as a group of one or multiple instances that collaborate among them by sharing their nodes in order to improve their respective utilities.

A cooperation strategy among instances is adopted, rather than a cooperation strategy among nodes, to prioritize the group interests in front of individual interests. Note that local decisions among nodes may also have an impact in the rest of the network. For example, if a node increases its transmission rate due to a collaboration with another node, it is also increasing the transmission rate of the nodes that forward its packets to the sink through the DODAG. This increases their energy consumption and it may even cause congestion problems in these nodes. Moreover, the traffic from the same instance can be generally aggregated thanks to its spatial and temporal correlation [27], but the traffic from other instances may contain different kind of measurements, not necessarily correlated or with different accuracy requirements. Since these cannot be efficiently aggregated, the energy consumption of the ascendant nodes is increased notably due to the additional traffic. Nevertheless, considering coalitions among nodes requires the implementation of highly distributed approaches, and therefore this remains as future work.

For the sake of clarity, in Table 1 we introduce the notation used in this article.

4.2. Rank computation and parent selection

Following the RPL approach, each node selects its parent node using the concept of rank. The main difference between both approaches in this sense is that in C-RPL nodes compute additional

Table 1
Symbols and notation.

n	Node id
i	Instance id
c	Coalition id
r	Rank
o	Objective function
e	Energy consumption
p	Parent node
u	Utility of a node
U	Utility of an instance
α	Cooperation parameter
$\mathcal{P}$	Set of possible coalitions
$\mathcal{O}$	Set of objective functions
$\mathcal{C}$	Set of nodes in coalition c
$\mathcal{I}$	Set of nodes in instance i
c^*	Best coalition
r^*	Best rank
e^*	Energy consumption associated with the best coalition
$\bar{a}$	Mean-square normalization of a

ranks in order to evaluate the possible coalitions that may be created in the network.

In general, the nth node defines its rank and parent related to each coalition and objective function, as

$$r^{*,c}_{n,o} = \min_{k \in \mathcal{C}} r^{k,c}_{n,o} \quad \forall c \in \mathcal{P}, o \in \mathcal{O}, \tag{1}$$

$$p^{c}_{n,o} = \arg\min_{k \in \mathcal{C}} r^{k,c}_{n,o} \quad \forall c \in \mathcal{P}, o \in \mathcal{O}, \tag{2}$$

where c is the coalition, $\mathcal{C}$ is the set of nodes that belong to this coalition, $\mathcal{P}$ is the set of possible coalitions in the network, o is the objective function, and $\mathcal{O}$ is the set of objective functions.

In networks with more than three instances, we limit the cooperation game to the coalitions between two instances, and the coalition between all the instances (referred to as the grand coalition). This is to avoid that the number of possible coalitions grows exponentially with the number of instances. Note that when the node density is low, the instances will tend to create the grand coalition in order to improve their performance. On the other hand, when the node density is high enough, the instances will tend to create small coalitions in order to reduce their energy consumption. This will be shown in Section 6.

Currently, DIOs in RPL have a unique *RPLInstanceID* and *Rank* fields, and this is not energetically efficient in multi-instance protocols. Note that in C-RPL all nodes need to broadcast the information associated with each possible coalition. Therefore, its signaling overhead related to DIOs would be multiplied by the number of objective functions and possible C-RPL Instances in the network. As a result, the implementation of this kind of protocols may not be feasible in many scenarios due to their high signaling cost. Nevertheless, the management of multiple instances has been identified as one of the open issues in RPL that needs to be addressed in the following versions [6]. Although this is out of the scope of this paper, in this paper we propose that the size of DIO packets should be more flexible, allowing nodes to include multiple *RPLInstanceID* and *Rank* fields in DIOs. Note with this simple modification, the nodes would not need to broadcast a different DIO for each objective function and possible C-RPL Instance, and thus the overall overhead compared to RPL would only be increased by the additional length of DIOs required to include the additional fields.

4.3. Coalition utilities

The creation of larger C-RPL Instances using coalitions may increase the performance of the network. However, coalitions are generally energy consuming, since they increase the overall idle consumption. Note that when two instances agree to collaborate, they both share their nodes. Since they have different duty cycles, the nodes from different instances must coordinate their active times in order to communicate. In a slotted listening duty cycle scheme, this means that either the transmitter or the receiver must remain active during the active time assigned to the other instance, increasing its duty cycle, and hence its idle consumption. We consider that the node that adapts its duty cycle in C-RPL is always the transmitter. On the other hand, the nodes forwarding packets from other instances increase their transmission rate because they have to forward additional packets. Although this may reduce the total traffic managed by the instance, and therefore its overall transmission consumption, in typical WSN applications the idle consumption is much higher than the transmission consumption (i.e., nodes remain in idle mode without transmitting most of the time). Therefore, we need to decide which coalitions, if any, are created based on their performance and energy consumption.

The performance of a node is characterized by its rank, and its energy consumption is estimated taking into account its duty cycle and average transmission rate in that coalition. In particular, we define the utility of the nth node in the coalition c, for a given objective function o_i, which is related to the instance i, as a combination of its rank $r^{*,c}_{n,o_i}$ and its associated energy consumption $e^{*,c}_{n,o_i}$ using linear aggregation as

$$u^{c}_{n,o_i} = -\left(\alpha \bar{r}^{*,c}_{n,o_i} + (1-\alpha)\bar{e}^{*,c}_{n,o_i}\right), \tag{3}$$

where $\bar{r}^{*,c}_{n,o_i}$ and $\bar{e}^{*,c}_{n,o_i}$ are the mean-variance normalization of $r^{*,c}_{n,o_i}$ and $e^{*,c}_{n,o_i}$, respectively. This is applied to homogenize the rank and energy consumption values. Note that the utility is inversely proportional to both rank and energy consumption. It is also worth noting that the nodes with a higher duty cycle, and therefore a higher $\bar{e}^{*,c}_{n,o_i}$, will be more prone to create coalitions and share their active time, since the impact of the additional consumption due to these coalitions is lower.

A weighted linear utility has been chosen, since it allows the network designer to easily adapt the willingness of nodes to increase their rank at the expenses of increasing their energy consumption using the parameter α. Although this parameter does not represent a physical metric, the mean-variance normalizations of the rank and energy consumption simplifies the selection of its value, which must be configured according to the application requirements (i.e., using $\alpha = 0.5$ nodes give the same importance to the changes in terms of rank and energy consumption related to a potential coalition). On one hand, low values will always tend to prioritize the consumption, while high values will tend to give more priority to the rank. In Section 6.2, we provide simulation results to guide the network designer in the configuration of this parameter.

Then, the instance i computes the utility of the coalition c as the minimum utility, when its associated objective function o_i is considered, among the nodes in $\mathcal{I}$ (i.e., set of nodes that belong to the instance i)

$$U^{c}_{i,o_i} = \min_{n \in \mathcal{I}} u^{c}_{n,o_i}. \tag{4}$$

Note that the minimum utility is considered to avoid unfair solutions among nodes. For example, solutions that overload a particular node.

4.4. Cooperation game among RPL instances

Once the utilities of each possible coalition have been computed, the instances must decide which coalitions are created in order to form the C-RPL Instances. We model instances as selfish entities that aim to maximize their performance with the minimum energy consumption. In this section, we show that without any coordination, the instances would never agree to cooperate among them.

Table 2
Prisoner's dilemma game.

		Player 2	
		Cooperate	Not Cooperate
Player 1	Cooperate	R_1, R_2	S_1, T_2
	Not Cooperate	T_1, S_2	P_1, P_2

Table 3
Cooperation game among RPL instances.

		I_2	
		Cooperate	Not Cooperate
I_1	Cooperate	U_1^c, U_2^c	U_1^c, U_2^n
	Not Cooperate	-	U_1^n, U_2^n

Following a game theoretical approach, we can model this game similarly to the prisoner's dilemma game [8] (Table 2). This game defines R "reward" as the utility when both players cooperate, S "sucker" as the utility when a player cooperates but the other player does not, T "temptation" stands for the utility when a player does not cooperate but the other player does. Finally, if neither cooperate, they both obtain a reward of P "punishment". The relation among them is $T > R > P > S$.

Comparing the prisoner's dilemma game with the cooperation game among two RPL Instances (Table 3), we can observe that they are very similar. In this game each instance is a player, where I_1 is the instance that requests the collaboration and I_2 is the instance that decides whether to collaborate or not. U^c is the utility when an instance cooperates (i.e., it shares its nodes), and U^n when it does not. Note that the *Not Cooperate - Cooperate* case is not considered here because this would mean that I_2 accepts a collaboration from I_1 that has not been requested.

Since the utility function considers both the performance and the energy consumption, U^c can be either higher or lower than U^n. However, analyzing this game, we can find that the Nash equilibrium is the *Not Cooperate – Not Cooperate* solution. This is because either if I_2 is willing to cooperate or not, I_1 always increases its utility by defecting to cooperate, standing that I_2 cooperates (i.e. the rank remains equal and the consumption is lower). As a result, even if I_2 had accepted to cooperate, it will also defect to cooperate because in this new situation its utility is higher if it does not share its nodes. Therefore, it is necessary to coordinate this process in order to find the best possible solution.

In order to decide which coalitions must be constructed, it is necessary the knowledge of the utility of each node in each possible coalition, and hence this decision cannot be distributed. In C-RPL, the sink node gathers the utilities computed by each node and computes the average utility of the network instances in each possible coalition. Then, it applies the cooperation game in order to find the best possible coalitions. Although the RPL protocol detects possible inconsistencies in the DODAG [6], in C-RPL the sink should also detect the inconsistencies among instances, which can be detected from the received packets. For example, if an instance in a coalition do not collaborate with the rest of instances in the same coalition, many packets of this coalition would not be received by the sink. In this case, the sink should compute and broadcast the new coalitions to the rest of nodes, and these should adapt their parent nodes accordingly.

4.5. Example of C-RPL

Finally, we present an illustrative example of C-RPL in a network with three instances, which manage three different kinds of traffic. In order to cover the main kinds of traffic that are present in WSNs, we consider event detection, non-critical monitoring and critical monitoring traffic. For each of these traffics, a different objective function is defined. In particular, in this example we consider the instances in Table 4.

Table 4
Instances coexisting in the network.

Instance	Traffic	Objective function
I_H	Event detection	H: Min. Hops
I_E	Non-critical monitoring	E: Min. ETX
I_P	Critical monitoring	P: Max. PDR

The computation of ranks is related to their objective function. Then, each node computes its own ranks as follows

$$r_{n,H}^{*,c} = \min_{k \in \mathcal{C}} r_{n,H}^{k,c} = \min_{k \in \mathcal{C}} \left(Hops_n^k + r_{k,H}^{*,c} \right) \quad \forall c \in \mathcal{P}, \tag{5}$$

$$r_{n,E}^{*,c} = \min_{k \in \mathcal{C}} r_{n,E}^{k,c} = \min_{k \in \mathcal{C}} \left(ETX_n^k + r_{k,E}^{*,c} \right) \quad \forall c \in \mathcal{P}, \tag{6}$$

$$r_{n,P}^{*,c} = \min_{k \in \mathcal{C}} r_{n,P}^{k,c} = \min_{k \in \mathcal{C}} \left(\frac{1}{PDR_n^k} r_{k,P}^{*,c} \right) \quad \forall c \in \mathcal{P}. \tag{7}$$

The value of $Hops_n^k$, ETX_n^k or PDR_n^k is estimated by the nth node using the DIOs coming from the kth node [6]. On the other hand, the values of $r_{k,H}^{*,c}$, $r_{k,E}^{*,c}$ or $r_{k,P}^{*,c}$ are broadcast by the kth node within its DIO packets. Note that ranks increase in the direction of the leaf nodes and decrease in the direction of the sink node.

Fig. 1 illustrates the possible coalitions evaluated by I_E. This instance evaluates the C-RPL Instance constructed using only its own nodes (Fig. 1a), the C-RPL Instances constructed in coalition with each other group: $\{I_E, I_P\}$ and $\{I_E, I_H\}$ (Fig. 1b), and also the C-RPL Instance that groups the whole network: $\{I_E, I_P, I_H\}$ (Fig. 1c). For each of these possible coalitions, the nth node in I_E computes a different rank:

$$r_{n,E}^{*,1} = \min_k r_{n,E}^{k,1} \quad \forall k \in I_E, \tag{8}$$

$$r_{n,E}^{*,2} = \min_k r_{n,E}^{k,2} \quad \forall k \in \{I_E, I_P\}, \tag{9}$$

$$r_{n,E}^{*,3} = \min_k r_{n,E}^{k,3} \quad \forall k \in \{I_E, I_H\}, \tag{10}$$

$$r_{n,E}^{*,4} = \min_k r_{n,E}^{k,4} \quad \forall k \in \{I_E, I_P, I_H\}. \tag{11}$$

On the other hand, the rest of the instances also evaluate all their possible coalitions. Then, the nodes in I_E compute some additional ranks, in this case using the metrics of the other instances. In particular, the rank using the nodes in I_E and the nodes of each of the other instances: $\{I_E, I_P\}$ and $\{I_E, I_P\}$ (Fig. 2a), and also the rank using the whole network considering each respective metric $\{I_E, I_P, I_H\}$ (Fig. 2b and c). Then, the nth node of I_E computes the following additional ranks:

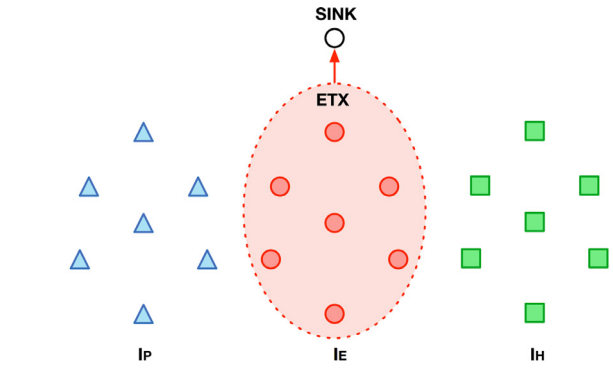
$$r_{n,P}^{*,5} = \min_k r_{n,P}^{k,5} \quad \forall k \in \{I_E, I_P\}, \tag{12}$$

$$r_{n,H}^{*,6} = \min_k r_{n,H}^{k,6} \quad \forall k \in \{I_E, I_H\}, \tag{13}$$

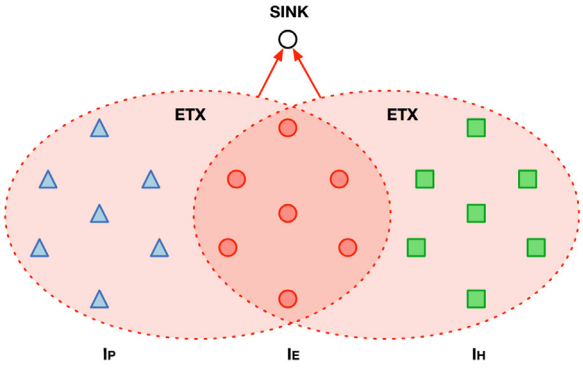
$$r_{n,P}^{*,7} = \min_k r_{n,P}^{k,7} \quad \forall k \in \{I_E, I_P, I_H\}, \tag{14}$$

$$r_{n,H}^{*,8} = \min_k r_{n,H}^{k,8} \quad \forall k \in \{I_E, I_P, I_H\}. \tag{15}$$

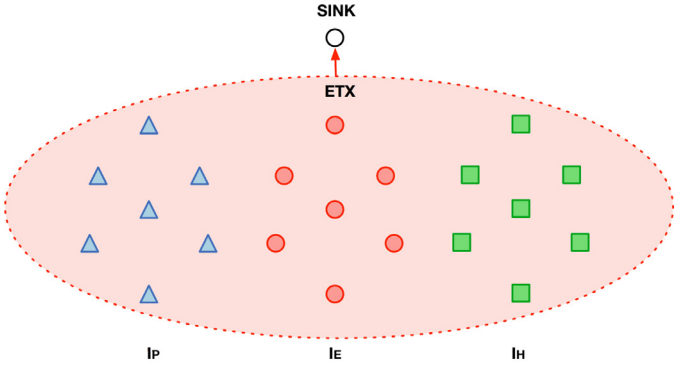
Summarizing, ranks $r_{n,E}^{*,1}$ to $r_{n,E}^{*,4}$ are used by I_E to evaluate the possible coalitions that can be formed with I_P and I_H, and are computed using the metric ETX. On the other hand, ranks $r_{n,P}^{*,5}$ to $r_{n,H}^{*,8}$ are used to inform the rest of instances about the performance that they would obtain by creating a coalition with I_E. Note that they are computed using the metrics of the other instances, which are Hops and PDR in this case.



(a) $r^{*,1}_{n,E}$

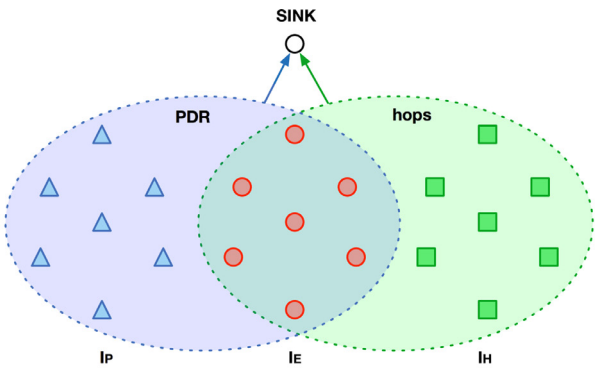


(b) $r^{*,2}_{n,E}, r^{*,3}_{n,E}$

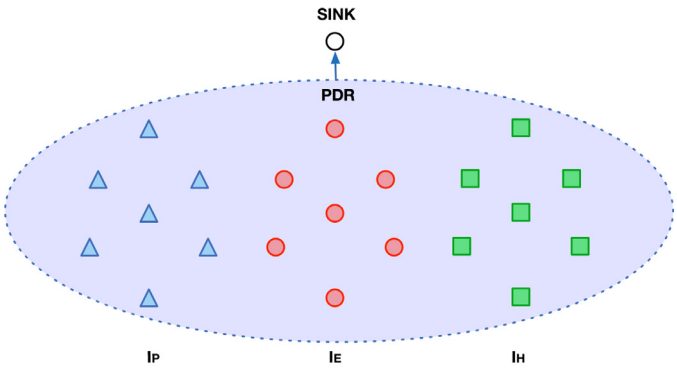


(c) $r^{*,4}_{n,E}$

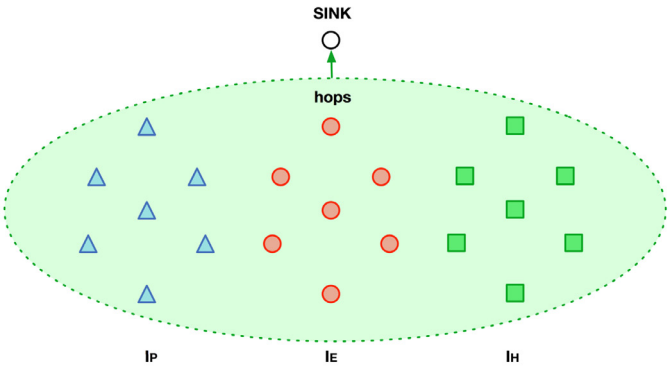
Fig. 1. Possible coalitions evaluated by I_E. Circles indicate the nodes available to construct the C-RPL Instance. The color of the circle indicates the rank metric used to evaluate the coalition. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) $r^{*,5}_{n,P}$



(b) $r^{*,6}_{n,H}, r^{*,7}_{n,P}$



(c) $r^{*,8}_{n,H}$

Fig. 2. Possible coalitions that involve I_E evaluated by other instances. Circles indicate the nodes available to construct the C-RPL Instance. The color of the circle indicates the rank metric used to evaluate the coalition. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5. Fairness in networks with multiple instances

The collaboration among instances can improve their performance, but it also increases their energy consumption. Without control, these collaborations may overload a particular instance, depleting the batteries of its nodes too fast. This situation may cause the loss of specific sensing capabilities. On the other hand, it is also important that these collaborations try to satisfy all their different requirements. Therefore, it is important that the available resources are shared in a "fair" manner among instances. Nevertheless, it is important to notice that the objective here is not to distribute the resources uniformly, since instances perform different tasks with different energy consumption demands.

In order to evaluate fairness, many definitions can be found in the literature [9] (e.g. weighted fairness, max–min fairness, proportional fairness). However, to the best of the authors knowledge, not any of these definitions has been used before to evaluate fairness in networks with multiple instances. In this paper, we propose to use the following metric

$$F = \min_{I} \left(\frac{1}{|\mathcal{I}|} \sum_{n \in \mathcal{I}} (x_n - x_n^*) \right) \quad \forall I \in \mathcal{K}, \tag{16}$$

where $\mathcal{K}$ is the set of instances in the network, x_n is the utility achieved by the nth node, and x_n^* is the optimum utility of this node. Since we evaluate fairness among instances, the average deviation of the instance' nodes is computed. The maximum value of

Table 5
Simulation parameters.

Parameter	Value
General	
Sensing area ($L \times L$)	150 × 150 m^2
Path loss at 1 m (PL_0)	50 dB
Path loss exponent (γ)	3
Flat fading variance (σ^2)	6
Packet size	127 bytes
Radio Transceiver	
Maximum data rate	250 kbps
Current consumption:	
Transmit state (I_{tx})	16.5 mA
Idle state (I_{idle})	15.5 mA
Sleep state (I_{sleep})	20 nA
Transmission power	3 dBm
Sensitivity	−91 dBm
Supply voltage (V_{DD})	3 V

fairness is found when $x_n = x_n^*$ $\forall n$. Then, fairness is always zero or negative.

In order to consider the tradeoff between performance and energy consumption, we evaluate fairness in both senses. In particular, the fairness in terms of rank (F_r) and energy consumption (F_e) are defined as follows

$$F_r = \min_i \left(\frac{1}{|\mathcal{I}|} \sum_{n \in \mathcal{I}} \left(\bar{r}_{n,o_i}^{*,*} - \bar{r}_{n,o_i}^{*,c^*} \right) \right) \quad \forall I \in \mathcal{K}, \tag{17}$$

$$F_e = \min_i \left(\frac{1}{|\mathcal{I}|} \sum_{n \in \mathcal{I}} \left(\bar{e}_{n,o_i}^{*,*} - \bar{e}_{n,o_i}^{*,c^*} \right) \right) \quad \forall I \in \mathcal{K}, \tag{18}$$

where c^* is the coalition selected by i, and $\bar{r}_{n,o_i}^{*,*}$ and $\bar{e}_{n,o_i}^{*,*}$ are the optimum mean-variance normalized rank and energy consumption of the nth node, respectively. These are the best values among all the possible coalitions in terms of rank and energy consumption.

This metric evaluates how the network resources are balanced among the instances in the network, taking into account the deviation of each node from its best possible solution. In particular, a very low value in F_r means that there is an instance that is not receiving enough resources, while a low value in F_e means that the network is overloading a particular instance. Since we want to avoid both extreme situations, it is important to evaluate the tradeoff between both metrics.

6. Simulation results

In this section, we compare the performance of C-RPL and RPL in a WSN with heterogeneous traffic using Matlab. This has been implemented in a centralized manner, since global knowledge is necessary to decide the network coalitions, but a distributed implementation of this algorithm remains as future work. On the other hand, we assume simple PHY and MAC layers to better observe the performance of C-RPL and RPL. Nevertheless, as we have explained in Section 4.1, similar conclusions can also be applied to networks using B-MAC [24] or ContikiMAC [26], which are the default LPL MAC protocols in TinyOS and Contiki, respectively.

We first evaluate C-RPL for different α values (i.e., cooperation parameter), node densities and traffic loads. Finally, we compare its fairness with RPL.

6.1. Simulation environment

In RPL, the nodes that may belong to each instance must be defined in advance. In this context, we consider two different versions of RPL, that construct their RPL Instances using different criteria. In the first version, simply referred to as RPL, RPL Instances may be composed of any node in the network, regardless of its specific tasks. In the second version, referred to as RPL II (Independent Instances), nodes belong only to the instance related to its tasks, and therefore to a unique RPL Instance. For the sake of comparison, we also include a second version of C-RPL, referred to as C-RPL NCG (No Cooperation Game), were each instance independently constructs the C-RPL Instance that maximizes its own utility. Note that this strategy is not equivalent to RPL, since this considers not only the rank, but also the energy consumption of each possible C-RPL Instance using (4). This has been included to show the lack of fairness that may arise without the cooperation game among instances.

Table 6
Traffics managed by the network.

Traffic	Duty cycle	Packets/s	Objective
Event detection	20 %	0.01	Min. Hops
Non-critical monitoring	5 %	0.1	Min. ETX
Critical monitoring	10 %	0.2	Max. PDR

We assume a very simple MAC layer that adjusts the duty cycle of each node according to its tasks in the network using a slotted listening duty cycling scheme (See Section 3). Then, a node must be active during the active time assigned to each of the instances in which it participates. For instance, a node that belongs to the instances that manage event detection and non-critical monitoring traffics would have a duty cycle of 25% (See Table 6). On the other hand, a TDMA-based scheme is used to avoid interferences among nodes from the same C-RPL Instance. Note that packet congestion may appear due to the limited bitrate of wireless sensors.

The simulation scenario considers a total of N sensing nodes, randomly deployed in a 150 × 150 m^2 area. The information collected by them is gathered at the sink node, that is located in the middle of the sensing area.

The channel losses (PL) are modeled following the log-distance path loss model [28]

$$PL(dB) = PL_0(dB) - 10\gamma log(d) + X, \tag{19}$$

where PL_0 is the path loss at the reference distance d_0 (1 m), γ is the path loss exponent, d is the communication distance, and X is the attenuation caused by flat fading, which has zero mean and variance σ^2. The values of these parameters have been empirically measured using Iris motes ($PL_0 = 50$ dB, $\alpha = 3$ and $\sigma_\gamma^2 = 6$). Moreover, we assume the sensitivity value (−91 dBm), the maximum transmission power (3 dBm) and the current consumptions in sleep, idle and transmit states (i.e., 20 nA, 15.5. mA and 16.5 mA, respectively) of the Iris transceiver (i.e., RF230). Table 5 summarizes the main simulation parameters.

In particular, the sink collects event detection, critical monitoring and non-critical monitoring information, coming from the sensor nodes, transmitting up to 250 kbps. The size of packets is assumed to be 127 octets (i.e., standard packet size in IEEE 802.15.4). Each kind of data is sensed by a different instance, which are composed of the same number of nodes. Since instances may sense different magnitudes, the traffic originated at different instances cannot be aggregated. On the other hand, the traffic from the same instance can be compressed using data aggregation techniques, since they may be spatially and temporally correlated. In particular, we assume a compression rate of 80% [27] (i.e. nodes can combine in average up to 5 packets into a single packet). The duty cycle, the average packet transmission rate and the objective function of each instance (See Table 6) are defined according to the particularities of their traffic. Note that these have different duty cycles and therefore their willingness to create coalitions is not the same. For example, the instance for event detection traffic will push more frequently to create coalitions in order to share

the active time of their nodes, while the instance for non-critical monitoring traffic will be less participative, since coalitions with the rest of instances increase notably its overall idle consumption.

6.2. Cooperation parameter

In Fig. 3, we show the impact of the cooperation parameter α in the network performance, in terms of the particular objective function of each instance. Although individual α values could have been considered for each node or instance, we assume a uniform α, for the sake of simplicity. We include the 95% confidence intervals in our simulations. In general, we can observe that instances do not cooperate until a minimum α is considered. This is because the network prioritizes the energy consumption in front of performance when using low α values. On the other hand, when the cooperation parameter is increased, C-RPL and C-RPL NCG tend to the RPL solution, since instances are more prone to collaborate with other instances, as long as they increase their performance. We can also observe that without the cooperation game, the instances start creating larger C-RPL Instances with lower α values. This is because these are decided individually. In this case, the best performance is always found using RPL, but in situations with congestion problems this may not be always the case, as we discuss in Section 6.4.

In Fig. 4, the average energy consumption is shown for different α values. As we have observed in Fig. 3, the network prioritizes the energy consumption using low α values. The average power consumption in C-RPL increases with α, from 5.5 mW to 8.5 mW, due to the additional communication among instances. This causes that more nodes increase their duty cycles, and therefore their energy consumption also increases. For the same reason, C-RPL NCG, has a higher consumption than C-RPL. In fact, this may be even higher than in RPL.

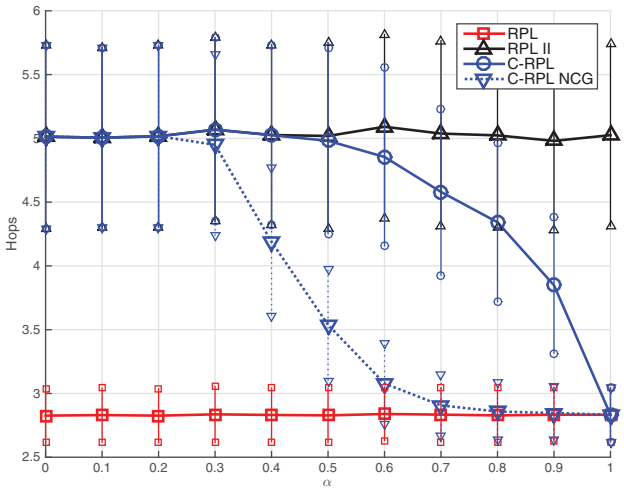
From Figs. 3 and 4, we have seen that we can manage the tradeoff between performance and energy consumption in C-RPL with α.

6.3. Impact of node density

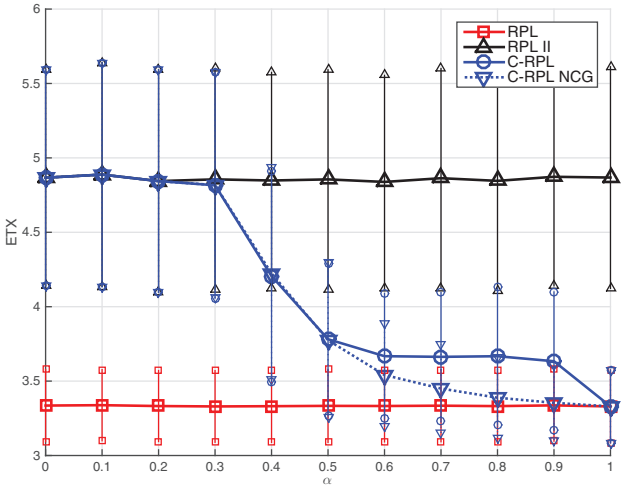
In Fig. 5, we evaluate the performance of RPL and C-RPL for different network densities, considering $\alpha = 0.9$. In networks with a low density of nodes, the best performance is achieved by RPL. On the contrary, when instances are independent from each other as in RPL II, the number of possible routes is reduced, and therefore their performance is lower. When the density of the RPL Instances increases, RPL II improves its performance, since the communication distances are shorter. In general, the performance of C-RPL is an intermediate value between both solutions. Note that the performance of each instance could be individually adjusted with individual α parameters.

As we can observe in Fig. 5c, a node density above 130 nodes generates network congestion in RPL, reducing the network performance. Note that I_P is the instance that generates the highest amount of traffic, and therefore it is congested at lower node densities than the rest of instances. However, the impact of node density is different in C-RPL, since C-RPL avoids creating C-RPL Instances with overloaded nodes. We discuss the network congestion problem in the next section.

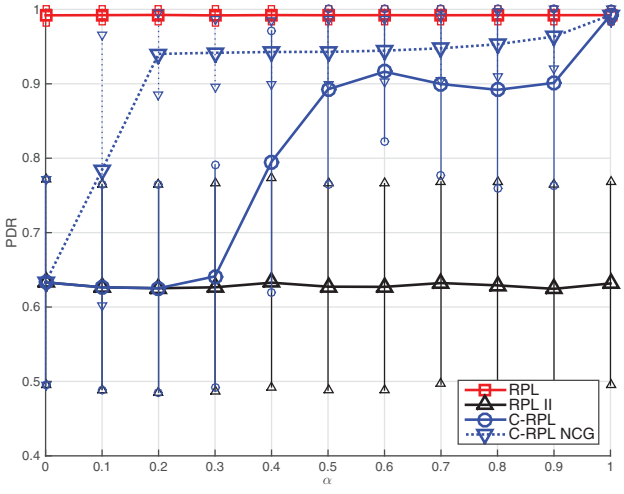
In general, C-RPL reduces the average number of coalitions with the node density, since instances do not need to create large C-RPL Instances to have a good performance. As a result, the average energy consumption of C-RPL can be always lower than RPL. This can be observed in Fig. 6, where C-RPL reduces the average power consumption of RPL around 17%.



(a) Average number of hops of I_H



(b) Average ETX of I_E



(c) Average PDR of I_P

Fig. 3. Performance in terms of hops, ETX and PDR for different α values. Each instance is composed of 40 nodes, which are randomly deployed in a 150 $\times$ 150 m^2 area.

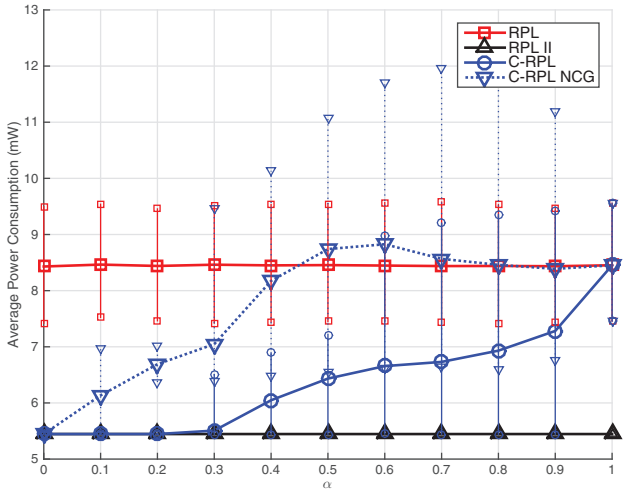


Fig. 4. Average energy consumption of the network for different α values. Each instance is composed of 40 nodes, which are randomly deployed in a 150 × 150 m^2 area.

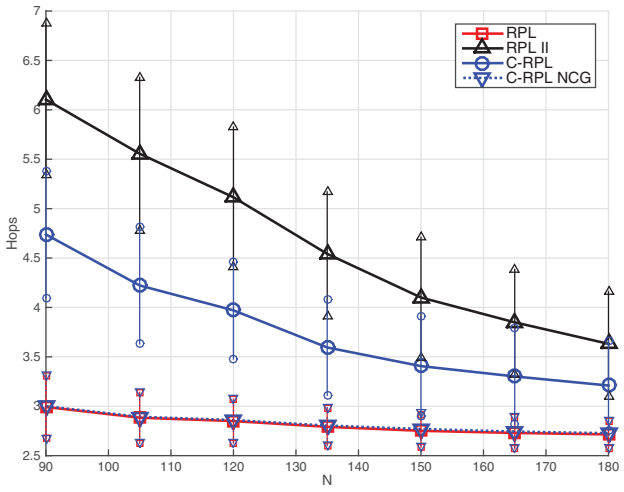
6.4. Impact of traffic load

The packet rates in the previous sections were not high enough to cause congestion problems with 120 nodes. However, bottlenecks may frequently appear in convergecast networks when the traffic is moderate, since the traffic is many-to-one. In this case, some nodes may not be able to forward all packets during their corresponding duty cycle. In Fig. 7, we show the impact of traffic congestion using different packet rates in I_P, while the packet rates of the rest of instances are not modified. We can observe that RPL is the most affected strategy due to the higher number of communications among nodes from different instances. As a result, a large amount of traffic cannot be aggregated, causing congestion problems in the network. For example, when each node in I_P generates 4 packets/s, the PDR is around 0.3. On the other hand, when instances are kept independent, the traffic can be always aggregated and therefore its impact is much lower (above 0.6 in the same case).

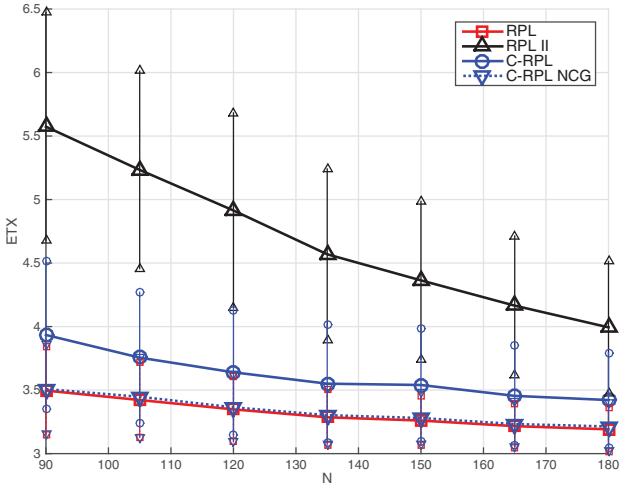
C-RPL manages this issue adjusting the cooperations according to the traffic conditions. When the traffic is low, the instances tend to form larger C-RPL Instances to achieve a better performance by collaborating among them. However, when the traffic is high enough to cause congestion problems, the C-RPL Instances tend to be smaller in order to reduce the amount of traffic that cannot be aggregated.

6.5. Fairness

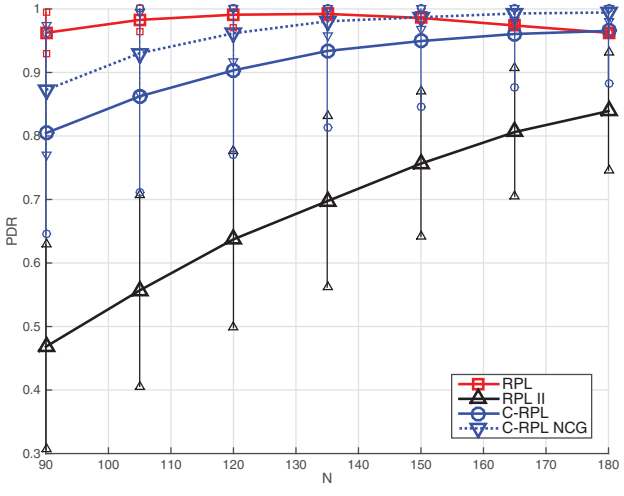
In applications with multiple sets of requirements, the network must not be focused in one specific task. Instead, it must distribute the available resources in a fair way to satisfy the requirements that each traffic demands. Fig. 8 shows the fairness of each strategy in terms of energy consumption and rank for different values of α. As we can observe, using RPL II the fairness in terms of consumption is close to zero, but the worst fairness in terms of rank is obtained (around −1.17). On the other hand, using RPL we can obtain a better fairness in terms of rank, but this is much worse in terms of energy consumption (around −1.45). We can also observe here that using C-RPL we can obtain intermediate solutions according to the α parameter. For instance, using $\alpha = 0.8$ C-RPL obtains a fairness around −0.75, both in terms of rank and consumption. Note that intermediate solutions mean that the resources are fairly shared, and any instance is overloaded.



(a) Average number of hops of I_H



(b) Average ETX of I_E



(c) Average PDR of I_P

Fig. 5. Performance in terms of hops, ETX and PDR for different node densities. Each instance has the same number of nodes (N/3). They are randomly deployed in a 150 × 150 m^2 area.

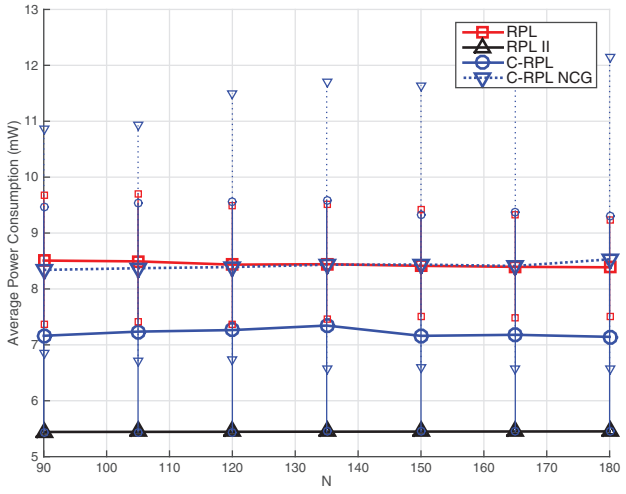


Fig. 6. Average energy consumption of the network for different node densities. Each instance has the same number of nodes ($N/3$). They are randomly deployed in a 150 × 150 m^2 area.

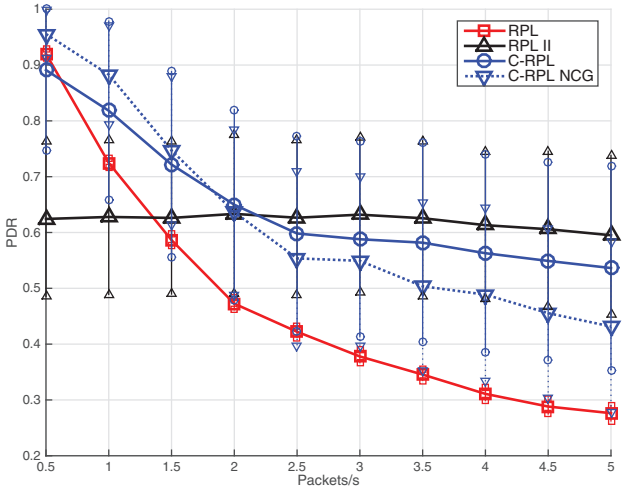


Fig. 7. Average PDR of I_P for different packet rates. Each instance has the same number of nodes ($N/3$). They are randomly deployed in a 150 × 150 m^2 area.

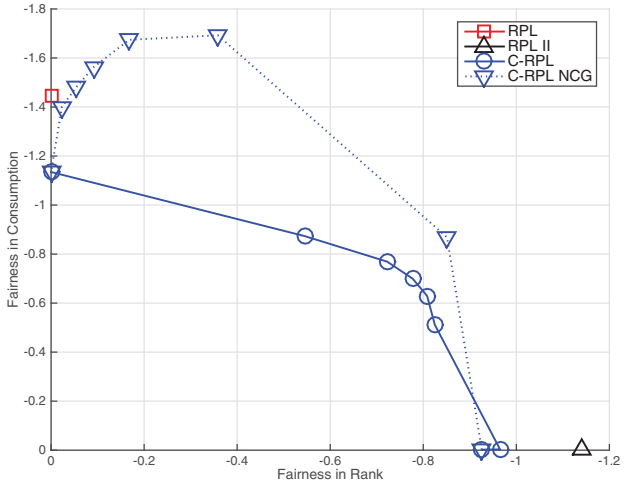


Fig. 8. Rank and consumption fairness for α=0, 0.5, 0.6, 0.7, 0.8, 0.9 and 1, where the smallest α is located on the right bottom side. Each instance is composed of 40 nodes, which are randomly deployed in a 150 × 150 m^2 area.

In this figure, we can also compare the fairness of C-RPL when the cooperation game is applied or not. In general we can observe that thanks to the cooperation game, C-RPL avoids solutions with a low fairness in terms of rank or energy consumption, while C-RPL NCG tends more to provide a good fairness in terms of rank, regardless of the fairness in terms of energy consumption.

7. Conclusion

Advanced WSN applications may need to sense, process and transmit data coming from different sensing units. As a result, different and maybe opposite sets of requirements must be satisfied in the same network. The well-known RPL protocol addresses this problem creating independent RPL Instances for traffics with different objective functions. However, it does not define the nodes that must belong to each instance, and this decision has a strong impact in the network energy consumption and performance. Therefore, in this paper we introduce C-RPL (Cooperative-RPL), that creates energy efficient instances, adapted to the particular QoS requirements of each traffic. These instances are constructed following a cooperative strategy among nodes with different sensing tasks. Since nodes would never agree to cooperate, as we have shown in this paper, C-RPL coordinates them to create coalitions that improve the tradeoff between their own performance metric and energy consumption, using the cooperation parameter α. The simulation results show that C-RPL efficiently creates the C-RPL Instances according to their individual objective functions and the particular network conditions. In particular, we have observed that C-RPL tends to create large instances when the node density is low. Otherwise, it constructs smaller instances to reduce the energy consumption of the network and avoid congestion problems. Besides, we have proposed a fairness analysis for networks with multiple instances that measures the distribution of the network resources when multiple objective functions coexist in the network. The results show that C-RPL obtains a more balanced fairness in terms of performance and energy consumption than RPL approaches with non-cooperative instances.

Acknowledgment

This work is supported by the Spanish Government under project TEC2014-53656-R and grant AP2010-1911.

References

[1] T. Harms, S. Sedigh, F. Bastianini, Structural health monitoring of bridges using wireless sensor networks, IEEE Instrum. Meas. Mag. 13 (6) (2010) 14–18, doi:10.1109/MIM.2010.5669608.
[2] M. Barcelo Llado, A. Correa Vila, J. Lo Vetere, A. Morell Perez, Cooperative multi-tree sleep scheduling for surveillance in wireless sensor networks, in: Proceedings of the IEEE Military Communications Conference, MILCOM, 2013, pp. 200–205, doi:10.1109/MILCOM.2013.43.
[3] D. Baghyalakshmi, J. Ebenezer, S. Satyamurty, Low latency and energy efficient routing protocols for wireless sensor networks, in: Proceedings of the International Conference on Wireless Communication and Sensor Computing, ICWCSC 2010, 2010, pp. 1–6, doi:10.1109/ICWCSC.2010.5415892.
[4] H. Luo, H. Tao, H. Ma, S. Das, Data fusion with desired reliability in wireless sensor networks, IEEE Trans. Parallel Distrib. Syst., 22 (3) (2011) 501–513, doi:10.1109/TPDS.2010.93.
[5] N.A. Pantazis, S.A. Nikolidakis, D.D. Vergados, Energy-efficient routing protocols in wireless sensor networks: A survey, IEEE Commun. Surv. Tutor. 15 (2) (2013) 551–591, doi:10.1109/SURV.2012.062612.00084.
[6] T. Winter, P. Thubert, A. Brandt, T. Clausen, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struick, J. Vasseur, RPL: IPv6 routing protocol for low-power and lossy networks, in: Proceedings of the IETF RFC 6550, 2012.
[7] P. de Melo, F. Cunha, A. Loureiro, A distributed protocol for cooperation among different wireless sensor networks, in: Proceedings of the IEEE International Conference on Communications (ICC), 2013, pp. 6035–6039, doi:10.1109/ICC.2013.6655566.

[8] R. Axelrod, W.D. Hamilton, The evolution of cooperation, Science 211 (4489) (1981) 1390–1396.
[9] H. Shi, R. Prasad, E. Onur, I. Niemegeers, Fairness in wireless networks:issues, measures and challenges, IEEE Commun. Surv. Tutor., 16 (1) (2014) 5–24, doi:10.1109/SURV.2013.050113.00015.
[10] B. Malakooti, I. Thomas, A distributed composite multiple criteria routing using distance vector, in: Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control, ICNSC '06., 2006, pp. 42–47, doi:10.1109/ICNSC.2006.1673115.
[11] H. Alwan, A. Agarwal, MQoSR: A multiobjective QoS routing protocol for wireless sensor networks, in: Proceedings of the ISRN Sensor Networks, volume 2013, 2013, p. 12, doi:10.1155/2013/495803.
[12] Y. Chen, N. Nasser, T.E. Salti, H. Zhang, A multipath QoS routing protocol in wireless sensor networks, Int. J. Sen. Netw. 7 (4) (2010) 207–216, doi:10.1504/IJSNET.2010.033204.
[13] H. Yetgin, K. Cheung, L. Hanzo, Multi-objective routing optimization using evolutionary algorithms, in: Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), 2012, pp. 3030–3034, doi:10.1109/WCNC.2012.6214324.
[14] P.V. Mieghem, F.A. Kuipers, On the complexity of QOS routing, Comput. Commun. 26 (4) (2003) 376–387, doi:10.1016/S0140-3664(02)00156-1.
[15] M. Arifuzzaman, M. Matsumoto, T. Sato, An intelligent hybrid MAC with traffic-differentiation-based QoS for wireless sensor networks QOS for wireless sensor networks, IEEE Sens. J., 13 (6) (2013) 2391–2399, doi:10.1109/JSEN.2013.2252163.
[16] T.-P. Chung, T.-S. Lin, X.-Y. Zheng, P.-L. Yen, J.-A. Jiang, A load balancing algorithm based on probabilistic multi-tree for wireless sensor networks, in: Proceedings of the Fifth International Conference on Sensing Technology (ICST), 2011, pp. 527–532, doi:10.1109/ICSensT.2011.6137035.
[17] W. Wei, A. Zakhor, Multiple tree video multicast over wireless ad hoc networks, IEEE Trans. Circuits Syst. Video Technol., 17 (1) (2007) 2–15, doi:10.1109/TCSVT.2006.885719.
[18] M. Barcelo, A. Correa, J. Lopez Vicario, A. Morell, Multi-tree routing for heterogeneous data traffic in wireless sensor networks, in: Proceedings of the IEEE International Conference on Communications (ICC), 2013, pp. 1899–1903, doi:10.1109/ICC.2013.6654799.
[19] G. Rajalingham, Y. Gao, Q.-D. Ho, T. Le-Ngoc, Quality of service differentiation for smart grid neighbor area networks through multiple RPL instances, in: Proceedings of the 10th ACM Symposium on QoS and Security for Wireless and Mobile Networks, in: Q2SWinet '14, ACM, New York, NY, USA, 2014, pp. 17–24, doi:10.1145/2642687.2642695.
[20] N.T. Long, M.-P. Uwase, J. Tiberghien, K. Steenhaut, QoS-aware cross-layer mechanism for multiple instances RPL, in: Proceedings of the International Conference on Advanced Technologies for Communications (ATC), 2013, pp. 44–49, doi:10.1109/ATC.2013.6698074.
[21] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, CTP: robust and efficient collection through control and data plane integration, Tech. Rep. SING-09-01, Stanford University, 2009.
[22] Z. Alliance, Zigbee specification version 1.0.
[23] W. Ye, J. Heidemann, D. Estrin, An energy-efficient mac protocol for wireless sensor networks, in: Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE INFOCOM 2002, vol.3, Vol. 3, 2002, pp. 1567–1576, doi:10.1109/INFCOM.2002.1019408.
[24] J. Polastre, J. Hill, D. Culler, Versatile low power media access for wireless sensor networks, in: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, in: SenSys '04, ACM, New York, NY, USA, 2004, pp. 95–107, doi:10.1145/1031495.1031508.
[25] M. Buettner, G.V. Yee, E. Anderson, R. Han, X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks, in: Proceedings of the 4th International Conference on Embedded Networked Sensor Systems, in: SenSys '06, ACM, New York, NY, USA, 2006, pp. 307–320, doi:10.1145/1182807.1182838.
[26] A. Dunkels, The ContikiMAC radio duty cycling protocol, Technical Report, SICS T2011:13 ISSN 1100-3154, 2011.
[27] C. Cappiello, F.A. Schreiber, Experiments and analysis of quality- and energy-aware data aggregation approaches in WSNs, in: Proceedings of the 10th International Workshop on Quality in Databases QDB, 2012.
[28] A. Martinez-Sala, J.-M. Molina-Garcia-Pardo, E. Egea-Ldpez, J. Vales-Alonso, L. Juan-Llacer, J. Garcia-Haro, An accurate radio channel model for wireless sensor networks simulation, J. Commun. Netw., 7 (4) (2005) 401–407, doi:10.1109/JCN.2005.6387982.