Task allocation in group of nodes in the IoT: a Consensus Approach

Giuseppe Colistra, Virginia Pilloni, Luigi Atzori
DIEE, University of Cagliari, Italy
{*giuseppe.colistra,virginia.pilloni,l.atzori*}@*diee.unica.it*

***Abstract*—The realization of the Internet of Things (IoT) paradigm relies on the implementation of system of cooperative intelligent objects with key interoperability capabilities. In order for objects to dynamically cooperate to IoT applications' execution, they need to make their resources available in a flexible way. However, available resources such as electrical energy, memory, processing, and object capability to perform a given task, are often limited. Therefore, resource allocation that ensures the fulfilment of network requirements is a critical challenge.**

In this paper, we propose a distributed optimization protocol based on consensus algorithm, to solve the problem of resource allocation and management in IoT heterogeneous networks. The proposed protocol is robust against links or nodes failures, so it's adaptive in dynamic scenarios where the network topology changes in runtime. We consider an IoT scenario where nodes involved in the same IoT task need to adjust their task frequency and buffer occupancy. We demonstrate that, using the proposed protocol, the network converges to a solution where resources are homogeneously allocated among nodes. Performance evaluation of experiments in simulation mode and in real scenarios show that the algorithm converges with a percentage error of about$\pm 5\%$ with respect to the optimal allocation obtainable with a centralized approach.

***Index Terms*—Consensus, resources allocation, Internet of Things.**

I. Introduction

The last few years have been involved by the technological revolution represented by the Internet of Things (IoT) [1]. The IoT vision aims to interconnect devices with different capabilities such as sensors, actuators, Radio Frequency Identification (RFID) tags, smart objects (e.g. smartphones), and servers, within the same heterogeneous network. The aim is to enable the network objects to dynamically cooperate and make their resources available, in order to reach a final goal, i.e. the execution of one or more applications assigned to the network.

Available resources (electrical energy, memory, processing, node capability to perform a given task) are often limited. This is the case, for example, of sensor nodes, which are often battery powered, and therefore have limited energy amounts. Another example is represented by the scarce processing capabilities of RFID tags. It is evident that an appropriate allocation of network resources would consistently improve network performance. Given the size of a distributed heterogeneous system such as the IoT network, the optimal resource allocation issue is not trivial. Resource allocation for IoT services is treated in [2][3]. In these studies, the aim is to find and allocate the resources that enable service execution. They do not focus on finding the best configuration that corresponds to an optimal resource allocation. To the best of authors' knowledge, there are not studies focusing on optimal resource allocation in IoT.

In this work, we propose a distributed optimization protocol based on consensus algorithm [4], to solve the problem of resource allocation and management. We also provide the semantic description of the middleware characteristics that would enable the implementation of this protocol in heterogeneous networks.

The rest of the paper is organised as follows. In Section II some preliminary considerations are drawn. Section III describes the reference IoT architecture. In Section IV, the consensus model in the algorithm is introduced. Section V describes the protocol. Finally, Sections VI and VII present the algorithm performance analysis and conclusions.

II. Past works

A. Resource allocation and management

Resource allocation has been extensively studied in the Wireless Sensor Network (WSN) field. A big effort has been put into resource allocation to extend WSN lifetime. In [5] the authors propose a framework to estimate network performance, which doesn't use node resources and does not introduce additional energy consumption that could compromise the network lifetime. In [6] a centralized task allocation algorithm to improve network lifetime is proposed. This study focuses on the reduction of the overall energy consumption into a heterogeneous WSN, with attention to nodes' residual energy. In [7] the same problem is analysed taking into account also task execution time. The authors in [8] propose the DLMA, an overlaying framework that determines the distribution of tasks among the nodes in a WSN by means of a distributed optimization algorithm, based on a gossip communication scheme, aimed at maximizing network lifetime. A similar approach is studied in [9], where a distributed algorithm based on particle swarm optimization is proposed.

As far as IoT networks are concerned, resource allocation is an open issue. Network heterogeneity, which regards both node capabilities and characteristic parameters, makes the resource allocation a challenging task. Semantic descriptions are needed, so that a common middleware can be designed in order to ensure the interoperability among different devices. Comprehensive ontologies that provide a semantic model for IoT are defined in [10] and [11].

Most of the existing studies on resource allocation for IoT are focused on IoT service provisioning, such as in [2] and [3]. None of the works found in the literature tries to find the optimal resource allocation associated to the lowest impact of the application assigned to the network. In this paper, a distributed optimization protocol, based on consensus algorithm, that solves the problem of resource allocation and management is described.

B. *Consensus protocol*

A consensus protocol is a collection of laws that regulates the interaction and the exchange of information between each node and its neighbours. All nodes in the network use the same algorithm in order to take decisions according to information available locally and received from other nodes. Olfati-Saber et al. [12] recall very well the history of consensus protocol from 1960 to recent years. The authors also describe as many seemingly different problems, which involve interconnection of dynamic systems in various areas of science and engineering, happen to be closely related to consensus problems. The applications where the consensus protocol has been used are various:

- *Synchronization of Coupled Oscillators* is important for several applications of engineering such as mobile target tracking, event detection, efficient scheduling, etc. [13][14][15].
- *Flocking Theory* is extensively studied by engineers because the coordination problem affects many applications such as massive distributed sensing using mobile sensor networks in an environment, self-assembly of connected mobile networks, automated delivery of payloads [4].
- *Rendezvous in Space* is equivalent to reaching a consensus in position by a number of agents. This problem is studied in the robotic field because using coordinated devices enables to perform a variety of challenging tasks, including search and recovery operations, surveillance, exploration and environmental monitoring [16].
- *Distributed Sensor Fusion in Sensor Networks* is the combination of sensory data from disparate sources such that resulting information is somewhat better than it would be when these sources are used individually. Consensus is used to coordinate nodes in the network [17].

III. THE REFERENCE SCENARIO

In the IoT, key nodes are represented by sensors, actuators, RFID tags, smart objects, and servers connected to the Internet [1]. In such a heterogeneous framework, nodes have the most diverse characteristics and capabilities: different residual energy, power consumption, processing capacity, available memory, and capability of performing a limited amount of tasks. In the reference scenario, all nodes need to interoperate in order to reason and allocate the available resources in a distributed way, with the aim of executing the application assigned to the network. Most of these decisions should be taken autonomously to avoid centralized solutions, which usually limit the flexibility of the systems and requires intense control data exchanges.

Most of the IoT powerful applications require the collaboration of different nodes, where each one performs a particular task and the mash-up of all the single tasks brings to the most disparate applications, e.g.: smart home monitoring, dangerous situation detection, tracking of goods, urban mobility assistant. In these scenarios it frequently happens that some nodes perform the same sensing operation, such as the measurement of the traffic in the same street, the measurement of the humidity and/or the temperature in a room, the detection of moving objects/persons in a given environment, the monitoring of the luminosity in a public square. Similarly, some other nodes may be interchangeable in the retransmission of data in a network when reaching the sink, typically in a wireless ad hoc network used to connect sensing nodes and the actuators to the wired Internet. Accordingly, the IoT is made of groups of nodes, i.e. *task groups*, that perform similar and replaceable tasks. These *task groups* are assigned with the relevant task by the application deployment server, which could decide which exact node should perform each needed task. Alternatively, it may leave these groups of nodes to autonomously decide how to distribute the burden of tasks among them without the need for the central server to keep the role of single physical node controller. According to this vision, the IoT is made of *virtual objects* (VO) [18] which are activated by the central deployment server. The VO role may be implemented by a node in the *task group* and is in charge of forwarding this signal to the other physical nodes (note that the virtual node may coincide with the only single physical node that is capable of implementing the required task). At this point, it is the duty of the nodes to allocate the proper resources to the required task.

Fig. 1 provides a sketch of the above described reference scenario. The central server transmits the activation signal to the VO that then forwards the information to the remaining nodes. The nodes are assumed to be geographically close each other. The aim of the algorithm explained in the following is, for each node belonging to the same *task group*, to dynamically change its assignment, in order to share the effort required to perform the considered task, in terms of necessary network resources. With the proposed protocol, nodes involved in the same task converge to the same task frequency and to the same local buffer usage. The idea is that, since the task is the same for all nodes, the accuracy of the application results will not be lost by balancing the task frequency among nodes.

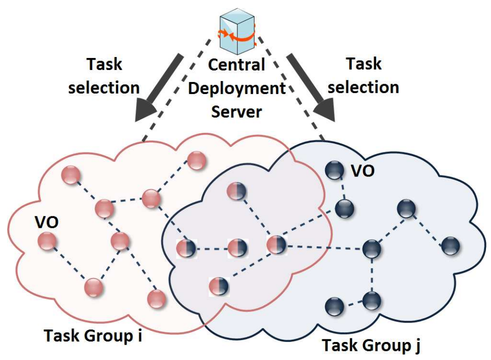

Fig. 1. The reference scenario.

As it is now evident in many other IoT scenarios, this framework requires the adoption of semantic technologies for the description of the objects capabilities. These are needed for the object to exchange information about their own capabilities and on the basis of this information to create the *task groups*.

The semantic description proposed in this paper is based on the ontologies presented in [10][11][19]. In particular, the Semantic Sensor Network (SSN) ontology is used to model sensor parameters, resources, services, and Quality of Service (QoS)/Quality of Information (QoI) related parameters. Fig. 2 shows an overview of the modules needed.

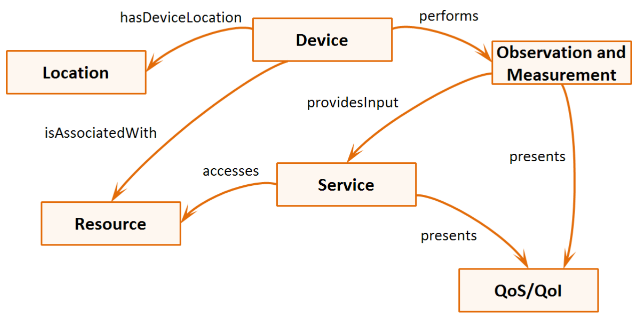

Fig. 2. Overview of the modules in the ontology.

Device: The Device module provides the description of device characteristics and capabilities. Devices can be sensors, actuators, processors, storage devices, tags, or a combination of these. Device resources are described in the Resource module. The Location module provides information about device position. Sensor devices are also related to the Observation and Measurement module.

Resource: The Resource module describes the type of resource associated to a device, and its related parameters. Since resources are needed for service execution, the Resource module is related to the Service module.

Service: The Service module provides an interface for the interaction between devices and their related processes. In particular, it defines how device resources are allocated to enable service execution, and which QoS/QoI characteristics are needed for that service. IoT services are modelled using the most widely used Web service languages, such as Unified Service Description Language (USDL), Web Service Definition Language (WSDL), and Web Application Description Language (WADL).

Observation and Measurement: The Observation and Measurement module describes how data are generated by sensor devices within the IoT scenario. Data are typically needed to provide services useful for the network application execution. When a certain QoS/QoI is required on measured data (e.g. a given data accuracy), the Observation and Measurements characteristics must enable the fulfilment of these requirements.

QoS/QoI: The QoS/QoI module defines the constraints that gathered measurements and provided services must fulfill. Since QoS/QoI is not always required, this is not a mandatory module.

Location: The Location module makes use of the GeoName ontology [20] to define device position. Since device position is not always needed, this is not a mandatory module.

IV. Model

A. Resource Model

Each node i that performs task k collects data with frequency $f_{i,k}$. The power consumed by node i is expressed by:

$$P_{i,k} = E_{i,k} \times f_{i,k} \tag{1}$$

where $E_{i,k}$ is the energy per task execution spent by node i for task k. Let N nodes perform task k. The total power consumption for task k:

$$P^c_k = \sum_{i=1}^{N} P_{i,k} \tag{2}$$

Similarly, the amount of data collected by node i at time t for task k is:

$$D_{i,k}(t) = B_k \times t \times f_{i,k} + M_i(t) \tag{3}$$

where B_k is the amount of output data for task k, and $M_i(t)$ is the occupancy of node i's storage buffer at time t. Therefore, the total amount of collected data due to task k at time t is:

$$D^c(t) = \sum_{i=1}^{N} D_{i,k}(t) \tag{4}$$

B. Consensus Model

We provide a mathematical model for ~~WSN~~ data. Every node i in a ~~WSN~~ has its own local data function, whose first order dynamic is given by:

$$\tau_i\,(t) = \alpha_i t + \beta_i \tag{5}$$

where τ_i is the number of samples collected by node i, α_i is the local task skew which determines the task frequency, and β_i is the local offset that describes the number of samples stored in the node buffer. We want to obtain the same virtual dynamic on all nodes. The virtual dynamic is defined by:

$$\tau_v\,(t) = \alpha_v t + \beta_v \tag{6}$$

Every node keeps an estimation of the virtual dynamic using a linear function of its own local function:

$$\tilde{\tau}_i = \tilde{\alpha}_i \tau_i + \tilde{o}_i \tag{7}$$

The goal is to find $\tilde{\alpha}_i$ and $\tilde{o}_i$ that compensate the difference among all node dynamics, and thus to converge to the virtual dynamic in Eq. (6). So, for each node i, the aim is to obtain that: $\tilde{\tau}_i \to \tau_v$. The consensus algorithm will be further explained in the following Section.

V. Protocol

In this work, we use the Protocol named Average TimeSync [13] based on two consensus algorithms. The first algorithm is used to compensate the skew. The second algorithm is used to compensate the offset. Protocol details are available in [13] and [14]. The implementation of the protocol includes three main parts: i. the relative skew estimation, ii. the skew compensation, and iii. the offset compensation.

A. Relative skew estimation

Each node in the WSN has a local counter, which stores the number of samples $\tau(t)$ collected since the start of the task. All nodes are capable to transmit their local counter $\tau(t)$. In the first step, every node i tries to estimate the relative skew with respect to its neighbours j as:

$$\alpha_{ij} = \frac{\alpha_j}{\alpha_i}$$

The value of α_{ij}, according to [13], can be estimated as:

$$\eta_{ij}^{+} = \rho_n \eta_{ij} + (1-\rho_n)\frac{(\tau_j(t_2)-\tau_i(t_2))}{(\tau_j(t_1)-\tau_i(t_1))} \tag{8}$$

where η_{ij} is the appraisal of relative skew and η_{ij}^{+} indicates the update of variable η_{ij} and $\rho_n \in (0,1)$ is a tuning parameter to compensate the noise. In [14] Theorem 4 demonstrates that $\lim_{t\to\inf} \eta_{ij}(t) = \alpha_{ij}$. Therefore, each node sends to its neighbours only the local counter $\tau(t)$. The related amount of data is very low, and it can be put inside another data packet producing a small overhead. From the point of view of the buffer occupancy, each node stores five variable for each neighbour: $\tau_j(t_2), \tau_i(t_2), \tau_j(t_1), \tau_i(t_1), \eta_{ij}$.

B. Skew compensation

In this Section, the first consensus algorithm is used to force all nodes to converge to a common task frequency defined in Eq.(6). The process is very simple. All nodes store their own virtual skew estimation defined in Eq.(7). As soon as node i receives a packet from node j, it updates the value of $\tilde{\alpha_i}$ according to:

$$\tilde{\alpha_i}^{+} = \rho_v \tilde{\alpha_i} + (1-\rho_v)\eta_{ij}\tilde{\alpha_i} \tag{9}$$

where ρ_v is a tuning parameter. For simplicity, in this work we set the initial condition as $\tilde{\alpha_i}(0) = 1$. The work in [13] demonstrates in detail that if Eq.(9) is expressed in matrix terms, it is a stochastic problem and that, according to stochastic matrix properties [12], we obtain $\lim_{t\to\inf} \tilde{\alpha_i}\alpha_i = \alpha_v$.

The previous demonstration entails some important remarks. The first is that the node transmission order is not important, nor is the exact moment the transmission takes place. So, this implies that the protocol is fully asynchronous and nodes can transmit at different rates. The only important condition is that the graph is sufficiently closely connected. Another important observation is that if any message is lost, the condition of strongly connected graph is still guaranteed. This implies that the algorithm is robust even against link failures, nodes failures and packet collisions. So the proposed protocol is very adaptive in dynamic scenario where the network topology change in runtime. From the performance point of view, with reference to transmitted and stored variables introduced previously, each node sends only the virtual skew estimation $\tilde{\alpha_i}$, which is a low amount of data. Also in this case, these parameters can be put inside a data packet, producing a small overhead. From the point of view of the node buffer, only one variable per each neighbour is needed: $\tilde{\alpha_j}$.

C. Offset compensation

In this Section, the first consensus algorithm is used to force all nodes to converge to a common offset of their dynamics. The compensation function, described in Eq.(10) is very similar to the skew compensation one:

$$\tilde{\beta_i}^{+} = \rho_b \tilde{\beta_i} + (1-\rho_b)(\tilde{\alpha_j}\tau_j + \tilde{\beta_j} - \tilde{\alpha_i}\tau_i - \tilde{\beta_i}) \tag{10}$$

where ρ_b is a tuning parameter. In [13] it is demonstrated in detail that $\lim_{t\to\inf} \tilde{\beta_i} + \frac{\alpha_v}{\alpha_i}\beta_i = \beta_v$. From Eq.(10) follows that this algorithm transmits only the virtual offset estimation $\tilde{\beta_i}$, and it stores only one variable for each neighbour: $\tilde{\beta_j}$.

D. Remarks on the Impact on Resources

Assuming that node i energy per task execution $E_{i,k}$ value (Eq.(1)) is the same for each node involved in task k, after the convergence of the proposed protocol, node i task frequency will be:

$$f_{i,k} = f_m = mean(\sum_{i=1}^{N} f_{i,k})$$

while the buffer occupancy will be:

$$M_i = M_m = mean(\sum_{i=1}^{N} M_i)$$

It follows that the total power consumption (Eq.(2)) and the total amount of collected data at time t (Eq.(4)) for task k have the same values as before the algorithm execution. Nevertheless, after the consensus protocol convergence the resources are equally shared by nodes involved in the same task.

VI. Performance Analysis

The performance analysis provides two cases studies: the first one is in the simulation scenario and the second one is the real scenario.

A. Simulation scenario

In this case study we used the Matlab software to implement a framework to simulate the protocol focusing on two types of communication: i. broadcast mode and ii. gossip mode.In both cases, the topology has been created following a random geometric distribution, and transmissions on the network are asynchronous. The broadcast communication entails that if the node i sends a packet, this is received by all neighbours, which update their values. On the other hand, the gossip mode entails that two nodes are selected in a pseudo-random way and communicate to update their values. The choice is pseudo-random because only two neighbours can communicate. Furthermore, for the simulation to be more realistic, we considered a certain probability of using a given link. We simulated a situation where the update values are inserted as data packet overhead. The simulation was run on 20 nodes (i.e. $N = 20$) in a random topology. We set all tuning parameters as: $\rho_n = \rho_v = \rho_b = 0.5$. We initialized node dynamics with random values of α and β. We assumed that nodes transmit a total amount of 5000 packets, so on average each node transmits 250 packets.

1) Broadcast communication: With this simulation we intended to study the performance of the protocol in terms of convergence speed and error, considering a broadcast communication among nodes.

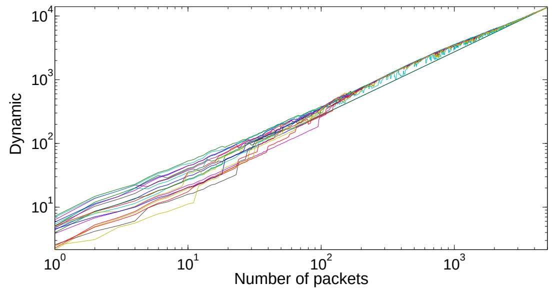

Fig. 3. Dynamics evolution using broadcast communication.

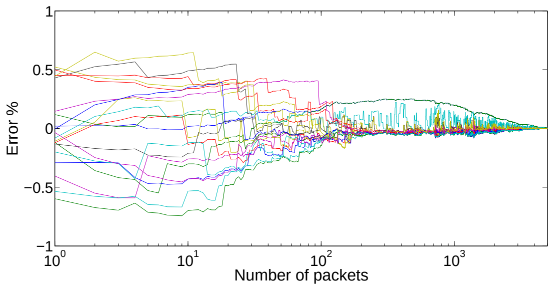

Fig. 4. Error evolution using broadcast communication.

Fig. 3 shows the algorithm convergence. After 100 packets transmitted on the network, (5 on average for each node) the dynamics can be considered converged. As the number of packets exchanged increases, nodes reach a better consensus. From the error point of view, Fig. 4 shows that initially the error is $\pm 60\%$, but after 100 packets are transmitted this value decreases by $\pm 20\%$, and eventually we obtain a very low error value of about $\pm 5\%$.

2) Gossip communication: With this simulation we intended to study the protocol performance in a more realistic scenario. Since the information exchanged to implement the protocol is very limited, as we explained in Section V, this information can be inserted in data packets. In this way, the algorithm's burden on the traffic network is low, as we can combine the consensus protocol with another application just used on the network.

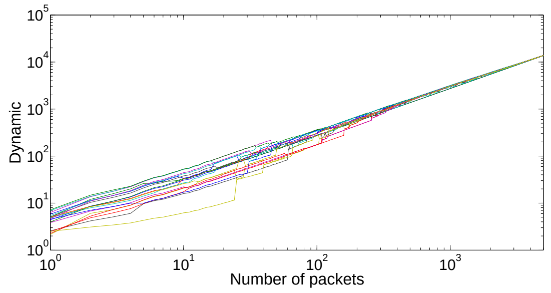

Fig. 5. Dynamics evolution by gossip communication.

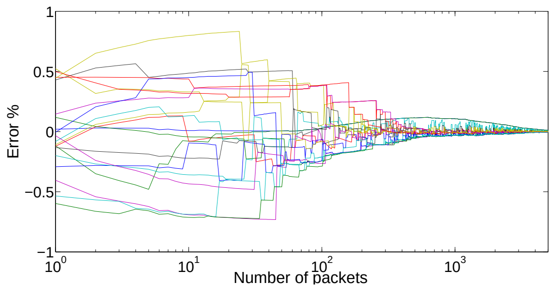

Fig. 6. Error evolution by gossip communication.

Fig. 5 shows the algorithm convergence. As in the broadcast communication scenario, nodes converge at consensus, but in this case convergence is slower. Fig. 6 shows this in more detail. When the packets transmitted on the network are about 100, the error is $\pm 40\%$, and this value decrease more slowly than in the broadcast communication scenario. This happens because in the gossip mode, in each iteration the communication is enabled only between two neighbouring nodes, so only these two nodes update their values. On the other hand, in the broadcast communication scenario all neighbours update their values simultaneously whenever a node transmits a packet.

B. Real scenario

The last experiment consists in the study of the protocol performance in a real scenario. In this section, we firstly illustrate the tools that we used. Then, we analyse the results to validate the performance of the proposed consensus application. The tools used for the experiments are the following:

- Development kit case provided by Telit Wireless Solutions. This kit is made of five ZigBee radio boards that are based on the Texas Instruments CC2530 System on Chip with the Embedded Telit Z-One ZigBee-PRO Stack. The antennas are external dipoles characterized by an omnidirectional pattern.
- The software used to inspect the packet content is Wireshark. To analyze the performance of the network from the Wireshark output and to conduct network discovery and commissioning, a specific tool named SRManager Tool has been developed by Telit Wireless Solution in collaboration with our lab. In this experiment, this tool has been used to set up the consensus protocol.

During the experiments we used three devices that communicated using the ZigBee standard on channel number 14 in the 2.4 GHz ISM frequency band. The type of communication is the gossip, because we inserted the necessary information for the well operation of the protocol as little overhead of the packets.

Fig. 7 shows the algorithm convergence. As in the simulation discussed in the previous Section, nodes converge at consensus. A good consensus has been reached after about 15 packets exchanged, corresponding to a mean of 5 update for node. From the error point of view, Fig. 8 shows that initially the error reaches peaks of $+80\%$ and -60%. Nevertheless, after 15 packets are transmitted, this value decreases by $\pm 20\%$, and eventually we obtain a very low error value of about $\pm 3\%$.

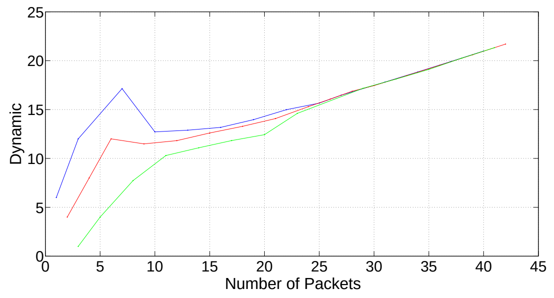

Fig. 7. Dynamics evolution by real scenario.

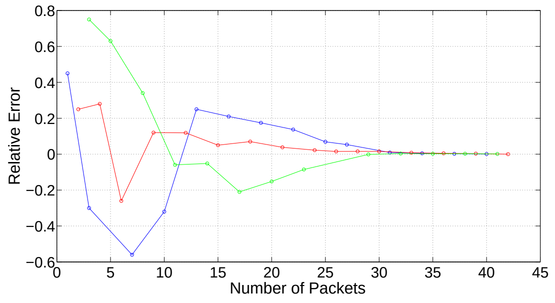

Fig. 8. Error evolution by real scenario.

VII. Conclusions

In this work, we applied a consensus protocol for an intelligent resource allocation in a IoT scenario. We propose a general IoT middleware designed using semantic languages that enables the use of the protocol in heterogeneous networks. We than described the adaptive resource allocation algorithm. We conducted simulated and real experiments. By applying the proposed protocol, we have been able to obtain a homogeneous resources allocation among all nodes in the network. The experiments also shown that the convergence of the proposed protocol strictly depends on the type of communication. We simulated two types of communication schemes for the exchange of update values among nodes during the algorithm execution: broadcast and gossip. In the broadcast case, convergence has shown to be faster than in the gossip one. In the broadcast case, the protocol allows to allocate resources with a percentage error lower than $\pm 20\%$ with respect to the reference value, in the case where each node transmits a mean of 5 packets, and decreases by $\pm 5\%$ when the number of packets transmitted increases. The case that uses gossip communication, by equal conditions, is slower to converge. When each node has transmitted a mean of 5 packets, we have a percentage error lower than $\pm 40\%$, which decreases by $\pm 5\%$. Finally, real experiments validate simulation results. Node dynamics has proved to converge, and percentage error value decreases by $\pm 20\%$, when each node has transmitted a mean of 5 packets, by $\pm 3\%$ when the number of transmissions increases.

Future works will be focused on the extension of the algorithm to the optimization of other important IoT resources, with attention to the fulfilment of QoS/QoI requirements. Furthermore, the proposed resource allocation algorithm can be easily combined with other resource saving mechanisms such as data fusion or data compression, that can be triggered by the algorithm to activate whenever needed.

References

[1] L. Atzori, A. Iera, and G. Morabito, “The internet of things: A survey,” *Computer Networks*, vol. 54, no. 15, pp. 2787 – 2805, 2010.

[2] D. Guinard, V. Trifa, F. Mattern, and E. Wilde, “From the internet of things to the web of things: Resource-oriented architecture and best practices,” in *Architecting the Internet of Things*. Springer, 2011, pp. 97–129.

[3] B. Silverajan and J. Harju, “Developing network software and communications protocols towards the internet of things,” in *Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE*. ACM, 2009, p. 9.

[4] R. Olfati-Saber, “Flocking for multi-agent dynamic systems: Algorithms and theory,” *Automatic Control, IEEE Transactions on*, vol. 51, no. 3, pp. 401–420, 2006.

[5] G. Colistra and L. Atzori, “Estimation of physical layer performance in wsns exploiting the method of indirect observations,” *Journal of Sensor and Actuator Networks*, vol. 1, no. 3, pp. 272–298, 2012.

[6] V. Pilloni and L. Atzori, “Deployment of distributed applications in wireless sensor networks,” *Sensors*, vol. 11, no. 8, pp. 7395–7419, 2011.

[7] J. Zhu, J. Li, and H. Gao, “Tasks allocation for real-time applications in heterogeneous sensor networks for energy minimization,” in *Proceedings of the Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, vol. 2, 2007, pp. 20–25.

[8] V. Pilloni, M. Franceschelli, L. Atzori, and A. Giua, “A decentralized lifetime maximization algorithm for distributed applications in wireless sensor networks,” in *Communications (ICC), 2012 IEEE International Conference on*, 2012, pp. 1392–1397.

[9] Y. Shen and H. Ju, “Energy-efficient task assignment based on entropy theory and particle swarm optimization algorithm for wireless sensor networks,” in *Green Computing and Communications (GreenCom), 2011 IEEE/ACM International Conference on*, 2011, pp. 120 –123.

[10] W. Wang, S. De, R. Toenjes, E. Reetz, and K. Moessner, “A comprehensive ontology for knowledge representation in the internet of things,” in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, 2012, pp. 1793–1798.

[11] S. De, T. Elsaleh, P. Barnaghi, and S. Meissner, “An internet of things platform for real-world and digital objects,” *Scalable Computing: Practice and Experience*, vol. 13, no. 1, 2012.

[12] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[13] L. Schenato and G. Gamba, “A distributed consensus protocol for clock synchronization in wireless sensor network,” in *Decision and Control, 2007 46th IEEE Conference on*. IEEE, 2007, pp. 2289–2294.

[14] L. Schenato and F. Fiorentin, “Average timesynch: A consensus-based protocol for clock synchronization in wireless sensor networks,” *Automatica*, vol. 47, no. 9, pp. 1878–1886, 2011.

[15] Z. Li, Z. Duan, G. Chen, and L. Huang, “Consensus of multiagent systems and synchronization of complex networks: a unified viewpoint,” *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 57, no. 1, pp. 213–224, 2010.

[16] J. Cortés, S. Martínez, and F. Bullo, “Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions,” *Automatic Control, IEEE Transactions on*, vol. 51, no. 8, pp. 1289–1298, 2006.

[17] L. Xiao, S. Boyd, and S. Lall, “A scheme for robust distributed sensor fusion based on average consensus,” in *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*. IEEE, 2005, pp. 63–70.

[18] J. Pascual Espada, Ó. Sanjuán Martínez, G. Pelayo, C. Bustelo, and J. M. Cueva Lovelle, “Virtual objects on the internet of things,” *IJIMAI*, vol. 1, no. 4, pp. 23–29, 2011.

[19] M. Compton, P. Barnaghi, L. Bermudez, R. Garcia-Castro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. Henson, A. Herzog *et al.*, “The SSN ontology of the W3C semantic sensor network incubator group,” *Web Semantics: Science, Services and Agents on the World Wide Web*, 2012.

[20] B. Vatant and M. Wick, “Geonames ontology (2006),” *Online at http://www. geonames. org/ontology.*