A Greedy Approach for Resource Allocation in Virtual Sensor Networks

Sonda Bousnina*, Matteo Cesana*, Jorge Ortín†‡ Carmen Delgado‡, José Ramón Gállego‡, María Canales‡

*Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy
{sonda.bousnina, matteo.cesana}@polimi.it

† Centro Universitario de la Defensa Zaragoza Academia General Militar, Zaragoza, Spain
jortin@unizar.es

‡Aragón Institute of Engineering Research Universidad de Zaragoza, Zaragoza, Spain
{cdelga, jrgalleg, mcanales}@unizar.es

***Abstract*—Virtual Sensor Networks (VSNs) envision the creation of general purpose wireless sensor networks which can be easily adapted and configured to support multifold applications with heterogeneous requirements, in contrast with the classical approach of wireless sensor networks vertically optimized on one specific task/service. The very heart of VSNs' vision is the capability to dynamically allocate shared physical resources (processing power, bandwidth, storage) to multiple incoming applications. In this context, we tackle the problem of optimally allocating shared resources in VSNs by proposing an efficient greedy heuristic that aims to maximize the total revenue out of the deployment of multiple concurrent applications while considering the inherent limitations of the shared physical resources. The proposed heuristic is tested on realistic network instances with notable performances in terms of execution time while keeping the gap with respect to the optimal solution limited (below 5% in the tested environments).**

I. Introduction

The last ten years have witnessed huge advancements in the field of Wireless Sensor Networks (WSNs) which have become one of the fundamental components to empower the vision of the Internet of Things (IoT). WSNs are nowadays ubiquitous in several applications environments like home automation, industrial control, healthcare, etc. Still the WSNs ecosystem has two main drawbacks which have slowed down the capillary diffusion of such technology: on one side, the WSN ecosystem is highly fragmented with multiple available alternatives at all the layers of the communication stack (PHY, MAC, network and application), on the other side the classical design approach is highly "vertical" with network deployments tightly customized on the needs of one or very few specific applications (*one-application/one-network* design approach).

In this context, there is an increasing need of orchestrating the diverse offer of communication protocols and standards, further being able to easily configure/re-configure the deployed WSNs to support multiple applications which might change over time as well. This view encompasses the creation of Virtual Sensor Networks (VSNs) in which multifold physical resources (sensor nodes, communication protocols, etc.) are virtually shared by multiple concurrent applications seamlessly. The vision of VSNs calls, on one side, for software platforms to abstract away the complexity and diversity of the available physical resources [1], [2], and, on the other side, it requires effective algorithms to dynamically allocate such physical resources to multiple incoming applications requesting for service.

This work focuses on the problem of optimal resource allocation in VSNs. Namely, we consider a wireless sensor network composed of heterogeneous physical resources in terms of sensor node hardware capabilities and communication protocols and we propose a simple yet effective algorithm to optimally deploy multiple applications while accounting for the constraints posed by the physical infrastructure. The proposed algorithm is then validated against the optimal solution to the resource allocation problem obtained in our previous work [3], which formalized the resource allocation problem as a Mixed Integer Linear Programming (MILP) and solved it with commercial solvers. The performance evaluation shows that the proposed algorithm is extremely fast when coping also with medium/large network scenarios, still providing solutions which are only 5% away from the optimal ones.

The rest of the paper is organized as follows: sections II and III respectively define the reference problem and introduce the proposed algorithm for resource allocation in VSN; Section IV reports and comments on the performance evaluation of the proposed solution. Related works on applying greedy-based heuristic to wireless networks are commented in Section V, while concluding remarks are finally given in Section VI

II. Reference Scenario and Problem Statement

We consider the same reference system as in [3] with a wireless sensor network composed of a set $S = \{s_1, s_2, \ldots, s_l\}$ of sensor nodes, and a set $A = \{a_1, a_2, \ldots, a_m\}$ of applications which need to be deployed in the wireless sensor network. Each application in set A has specific "sensing" requirements, defined as specific positions in the environment which need to be monitored (e.g., one application may require to measure scalar parameters like the temperature at specific points/areas). Formally, each application j has to sense a given set of test points T_j, which requires application j to be deployed on a subset of S such that all the test points in T_j are covered. A test point is covered by a sensor node i if it is within its

sensing range, R_i^s. Based on this, we can define the set S_{jk} as the set of sensor nodes which physically cover the test point k, with $k \in T_j$. In other words, if the application j is deployed on a sensor in set S_{jk}, then the test point k is covered for this application. We consider here the most demanding case in which an application to be successfully deployed needs to cover all the test points in its set T_j; the proposed framework can easily be extended also to the case where partial coverage is needed. For instance, let us assume that application j has 2 test points, $T_j = \{t_1, t_2\}$. We can further assume that test point t_1 is covered by sensor nodes $S_{j1} = \{a_1, a_2\}$, and test point t_2 is covered by sensor nodes $S_{j2} = \{a_3, a_4, a_5\}$. Then, in order to deploy application j on the network, it must be deployed in either a_1 or a_2, and also in a_3, a_4 or a_5.

Moreover, each application j in A is further characterized by a requirement vector $r_j = \{c_j, m_j, p_j\}$ which specifies the required source rate [bit/s], memory [bits] and processing load [MIPS] consumed by the application when it is deployed on a sensor node. Dually, each sensor node i in S is characterized by a given resource vector $o_i = \{C_i, M_i, P_i, E_i\}$, which specifies its available bandwidth, storage capabilities, processing power and energy store.

The information generated by the applications deployed at sensor nodes needs to be delivered remotely to sink nodes through multihop paths which are computed by operating any state-of-the-art routing protocol for wireless sensor networks; without loss of generality, we consider here the RPL [4] as the reference routing protocol, being the routing metric the hop count measure. We denote by P_i the path from a given node s_i to its closest sink and with $h(s_i)$ its length. The path is formed by a concatenation of links $s_i \rightarrow s_h \rightarrow s_g \rightarrow \ldots \rightarrow s_{\text{sink}(i)}$, with $s_h \rightarrow s_g$ denoting the link between the node s_h and the node s_g. We call l_h the link $s_h \rightarrow s_{s(h)}$, where $s_{s(h)}$ is the successor of s_h in the shortest path tree calculated by the reference routing algorithm. We also indicate that the link l_h belongs to P_i by writing $l_h \in P_i$.

A protocol interference model with power control is used to capture wireless interference between nodes, that is, the transmission power of each node is set to the minimum level required to reach the receiver node and the transmission capacity of a given wireless link is shared by all the concurrent transmissions in the interference range of the reference link. Formally, the *interference set* of link l_h, I_h, is the set of links that interfere or are interfered by link l_h. This set is formed by the links whose receiver is within the interference range of the node s_h and the links where the node $s_{s(h)}$ is within the interference range of its transmitter. Every time the link l_h is transmitting, the links in I_h must remain silent so that there is no interference between them (and conversely, every time a link in I_h is transmitting, l_h must be silent). Therefore, the transmission time of link l_h is shared among all the links in I_h.

We aim to maximize the overall profit of deploying applications in the VSN which can be measured as the revenues obtained in succeeding deploying an application in the VSN discounted by the cost incurred in terms of needed physical resources; namely, the reference objective function is:

$$\max \left(\sum_{j \in A} q_j z_j - \sum_{i \in S} \delta_i x_i \right) \tag{1}$$

where q_j is the revenue for deploying application j, z_j is a binary variable indicating if application j is deployed, δ_i is the cost of activating sensor node i and x_i is a binary variable indicating if node i is active.

The reference optimization problem scales down to find which applications to deploy and where to deploy them in order to maximize Eq. (1), while matching the following constraints set on the physical network infrastructure: (i) *resource constraints*, the applications must be deployed at sensor nodes which "enough" physical resources (memory, processing capacity, energy and bandwidth), (ii) *capacity constraints*, there is "enough" bandwidth to deliver the data flow generated by the deployed applications to the remote sinks across multi-hop paths.

III. Greedy Algorithm

The core idea of the proposed algorithm is to sort all the applications in set A in decreasing order with respect to a *profit ratio* e_j, which measures the "profit gain" that is obtained if application j is successfully deployed. The algorithm proceeds by selecting the application with the highest e_j and try to deploy it by checking if all the resource and bandwidth constraints are verified. The application (deployed or not) is removed from set A and the process is then iterated until set A is empty. The *profit ratio* is defined as:

$$e_j = q_j / w_j, \tag{2}$$

being q_j the revenue for deploying application j and w_j an estimation of the cost of deploying the application. Such cost is:

$$w_j = \sum_{k \in T_j} w_{jk} \tag{3}$$

where w_{jk} is the cost incurred in covering the k-th test point of application j. This cost depends on the specific sensor node i chosen to cover the k-th test point of application j, which we define as v_{ji}. In order to consume as few resources as possible, we assume that the application will be deployed on the node with the lowest v_{ji} among those nodes with enough spare resources to host it, that is,

$$w_{jk} = \min_{i \in S'_{jk}} v_{ji} \tag{4}$$

with S'_{jk} the set of nodes that cover test point k of application j and that have enough resources to sense effectively this test point. This implies that the nodes in S'_{jk} has enough available memory, processor capacity and energy to sense and transmit the sensed data. They must also have a path to the sink with enough transmission resources and energy to retransmit the sensed information.

We consider two different alternatives for measuring the cost factor v_{ji}. In the first case, we set $v_{ji} = h(s_i)$, i.e. the number

of hops to reach the sink from the node. In the second case, we define v_{ji} as

$$v_{ji} = \sum_{l_h \in P_i} \frac{c_j}{C_h} |I_h| \quad (5)$$

This expression is a measure of the total transmission resources used in the network when the application is deployed on the node s_i: the ratio c_j/C_h is the percentage of airtime required to (re)transmit the data from application j by sensor node h. As this airtime is also consumed for all the links that interfere with the link h, this ratio is multiplied by the number of elements in I_h. Finally, this amount is added up for all the links that form the path to the sink.

The detailed pseudocode of the proposed solution is reported in Algorithm 1. First, we define for each node the variables $M_i^{(r)}, P_i^{(r)}$ and $E_i^{(r)}$, which are used to track the available memory, processing capacity and energy of the nodes as applications are deployed on the network. We also define for each link the variable $B_i^{(r)}$, which is used to store the available transmission resources of that link. Variables $M_i^{(r)}, P_i^{(r)}$ and $E_i^{(r)}$ are initialized to M_i, P_i and E_i, while variables $B_i^{(r)}$ are set to 1 to indicate that the whole airtime of each link is fully available at the beginning. We also initialize each set S'_{jk} with all the elements in the corresponding set S_{jk} since we assume at first that any application fits into any node that can sense its test points (lines 1-2).

Next, the terms e_i are computed for all the applications in A with eqs. (2)-(5) (line 3) and the algorithm enters into its main loop. At each iteration, we look for the application with the highest value of e_j, (application a_b in line 5), and try to deploy it on the network. For simplicity, this part of the algorithm is explained in Algorithm 2. First, we define the auxiliary variables $M_i^{(a)}, P_i^{(a)}, E_i^{(a)}$ and $B_i^{(a)}$, which are used to store a copy of $M_i^{(r)}, P_i^{(r)}, E_i^{(r)}$ and $B_i^{(r)}$. These variables allow obtaining the remaining resources of the network if a_b can be effectively deployed (i.e. if there are enough resources in the network).

To check this, we have to verify that all the test points of a_b can be properly sensed (line 2). To that end, we try to sense each test point with the sensor node s_f that has the lowest associated cost v_{bi} (line 3), calculating: (*i*) the remaining available memory and processor capacity that node s_f would have if the application were deployed on it (lines 4-5), (*ii*) the remaining transmission resources of each link of the path P_f (line 8) and of the links in their interference sets (line 10), (*iii*) the remaining energy in s_f and in all the nodes of the path P_f (line 7). For each link, the required transmission resources correspond to the airtime needed to transmit the sensed data, which is c_b/C_g, with c_b the bit rate of application a_b and C_g the available bandwidth of the node. Regarding the energy consumption, it depends on the application type and on the sensor node type and its role (if it senses and transmits the data or if it only receives and retransmits it). This energy consumption is left as a function of a_b and s_g.[1]

[1]Detailed expressions for this energy consumption can be found in [3].

Algorithm 1 Scheme of the proposed solution

1: Initialize variables $M_i^{(r)}, P_i^{(r)}, E_i^{(r)}, B_i^{(r)}$
2: Initialize sets S'_{jk}
3: Compute e_j with (2)-(5) for all $a_j \in A$
4: **repeat**
5: $a_b \leftarrow \arg\max_{j \in A}(e_j)$
6: **if** a_b can be deployed **then**
7: Update variables $M_i^{(r)}, P_i^{(r)}, E_i^{(r)}, B_i^{(r)}$
8: Remove application a_b from A
9: **else**
10: Remove from S'_{bk} the node s_f that cannot sense test point t_k of application a_b
11: **if** S'_{jk} is empty **then**
12: Remove application a_b from A
13: **else**
14: Recompute e_b with (2)-(5)
15: **end if**
16: **end if**
17: **until** A is empty
18: Compute objective function with (1)

If all the auxiliary variables $M_i^{(a)}, P_i^{(a)}, E_i^{(a)}$ and $B_i^{(a)}$ are higher than zero once all the test points of application a_b are covered, we can state that a_b can be deployed on the network (line 18 of Algorithm 2). If so, we remove a_b from the set A and update the variables $M_i^{(r)}, P_i^{(r)}, E_i^{(r)}$ and $B_i^{(r)}$ with $M_i^{(a)}, P_i^{(a)}, E_i^{(a)}$ and $B_i^{(a)}$ (lines 7-8 of Algorithm 1). If it cannot be deployed (because some test point cannot be covered with the node with the lowest cost), we remove that node from the set S'_{bk} (line 10 of Algorithm 1). If this set is empty, we remove the application from A as it is not possible to cover test point t_k (lines 11-12). If S'_{bk} is not empty, the application might still be deployed, so we update e_b and go back to line 5. Finally, when the set A is empty, we compute the objective function with (1).

IV. Simulations and performance evaluation

In this section, we evaluate the performance of the proposed solution with the two different definitions of v_{ji}: $h(s_i)$ (**GreedyHop**), and Eq. (5) (**GreedyAirtime**). We also consider the case of running the algorithm twice (one with each definition of v_{ji}) and taking the best result (**GreedyMax**). We compare the results with the optimum solution (**Optimum**) given in [3] obtained by formalizing the resource allocation problem as a MILP and solving it via commercial solvers; optimality gap and computation time are the main performance metrics used in the evaluation.

A. Simulation Environment

We consider a scenario with two different types of sensor node hardware and two classes of applications. In the following, we define the main features of both sensor nodes and applications and the simulation parameters. Then, results are presented.

Algorithm 2 Check if an application fits into the network

Initialize variables $M_i^{(a)}, P_i^{(a)}, E_i^{(a)}, B_i^{(a)}$
for all $t_k \in T_b$ **do**
  $s_f \leftarrow \arg\min_{i \in S'_{bk}}(v_{bi})$
  $M_f^{(a)} \leftarrow M_f^{(a)} - m_b$
  $P_f^{(a)} \leftarrow P_f^{(a)} - p_b$
  for all $l_g \in P_f$ **do**
    $E_g^{(a)} \leftarrow E_g^{(a)} - e(a_b, s_g)$
    $B_g^{(a)} \leftarrow B_g^{(a)} - c_b/C_f$
    for all $l_h \in I_g$ **do**
      $B_h^{(a)} \leftarrow B_h^{(a)} - c_b/C_f$
    end for
  end for
  if any variable $M_i^{(a)}, P_i^{(a)}, E_i^{(a)}, B_i^{(a)}$ is < 0 **then**
    a_b does not fit into the network
    return s_f
  end if
end for
a_b fits into the network
return $M_i^{(a)}, P_i^{(a)}, E_i^{(a)}, B_i^{(a)}$

1) Sensor Node Hardware: The network topologies used for performance evaluation include two heterogeneous sensor nodes platforms with different characteristics in terms of cost and available resources: (i) *basic* highly-constrained hardware whose parameters have been derived by taking as a reference the TelosB sensor platforms [5]; (ii) *high-level visual* sensor nodes well represented by BeagleBone platforms [6]. Table I summarizes the main characteristics of these platforms [7].

TABLE I
SENSOR NODES.

	Basic	**High-Level**
Reference Hardware	Telos-B	BeagleBoard
TX Rate (C_i)	250[kb/s]	250[kb/s]
Available RAM (M_i)	7[Kbyte]	256[Mbyte]
Processing Rate (P_i)	8[MIPS]	720[MIPS]
Energy Store (E_i)	3200[J]	3200[J]

2) Applications: Two classes of applications are considered: *scalar* and *visual* applications. Scalar applications require the collection and delivery of scalar information like temperature and luminance samples. *Visual* applications [8] require the collection, processing and delivery of multimedia content (images and video). While *scalar* applications can be supported by both types of hardware platforms, *visual* applications can only be deployed at high-level sensor nodes. For *visual* applications we focus on visual sensor networks (e.g. object recognition) [9]. We consider two paradigms to perform visual tasks: the Compress-Then-Analyze (CTA) and the Analyze-Then-Compress (ATC) [9], [10]. Tables II and III summarize the characteristics of the reference applications [7].

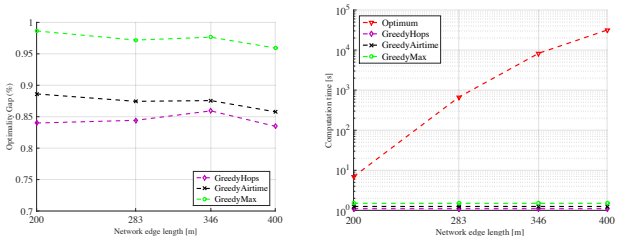
3) Scenario Topologies: The results presented hereafter have been obtained on 3.0 GHz Quad Core Intel Woodcrest (64 bits) machine with 8 GB RAM and 250 GB SATA storage, averaging over 100 randomly generated network topologies for each scenario as detailed in Table IV.

TABLE II
SCALAR APPLICATIONS.

	Temperature Mon.	**Light Mon.l**
Generated Data (c_j)	0.5[kb/s]	1[kb/s]
Bytecode footprint (m_j)	4462[byte]	1006[byte]
Available RAM (l_j)	negligible	negligible

TABLE III
VISUAL APPLICATIONS.

	CTA	**ATC**
Generated Data (c_j)	20[kb/s]	12[kb/s]
Bytecode footprint (m_j)	10[kbyte]-256[Mbyte]	10[kbyte]-256[Mbyte]
Available RAM (l_j)	17.64[MIPS]	69.23[MIPS]



(a) (b)

Fig. 1. Performance Evaluation. a) Optimality gap (ratio between the objective function defined in Eq. (1) calculated by the reference heuristic and the optimal value; b) Computation Time (in log scale)

TABLE IV
SCENARIO TOPOLOGIES.

	Scenario			
	1	2	3	4
Size	200 × 200 m	283 × 283 m	346 × 346 m	400 × 400 m
# Nodes	36+36	72+72	108+108	144+144
# sinks	1+1	2+2	3+3	4+4
# applications	6+6+6+6	12+12+12+12	18+18+18+18	24+24+24+24

B. Evaluations and Results

Figures 1(a) and 1(b) summarize the performance of the proposed algorithm in terms of optimality gap and processing time, when increasing the scale of the reference network scenario (see Table IV). As clear from the figures, the proposed solutions are characterized by negligible computation time with respect to the case where the problem is formalized as a MILP and solved at the optimum with commercial solvers; yet, the optimality of the most effective heuristic (**GreedyMax**) is often within 5% in all the tested network topologies.

The objective function (1) is composed of two contrasting terms, the first favoring solutions with higher numbers of deployed applications, the second favoring "cheaper" networks (with fewer active nodes); in this view, it is worth analyzing the behavior of the proposed heuristics as far as the two

terms of the objective function are concerned. Figures 2 and Figures 3 report the number of deployed applications and the corresponding cost (number of sensor nodes) for different network topologies; looking at figures 2.a and 3.a, one can observe that the **GreedyAirtime** solution is able to deploy a number of applications comparable to the optimum case, but, on the other hand, it results in more expensive network infrastructures; this is mainly due to the fact that the **GreedyAirtime** is "biased" in deploying applications in those sensor nodes which are less loaded and interfered (see Eq. (5)); this trend is verified also by Figures 3(b) and 3(c) which report the type of activated sensor nodes.

The **GreedyHop** algorithm provides the cheapest network infrastructures with the lowest number of deployed applications, whilst the **GreedyMax** strikes a better balance between number of deployed applications and network cost. Figures 2(b) and 2(c) report the breakdown of the deployed applications distinguished in scalar and visual ones.

Tables V and VI analyze the sensitivity of the proposed algorithms with respect to the cost for activating sensor nodes; namely, the results report the number of deployed applications and activated nodes when varying parameter δ of the objective function in Eq. (1) for the largest network instances (Scenario 4 of Table IV). Expectedly, when increasing the activation cost for sensor nodes, the proposed algorithms all tend to trade off the number of deployed applications (revenues) to reduce the infrastructure cost.

TABLE V
NUMBER OF DEPLOYED APPLICATIONS WHEN VARYING THE COST PARAMETER δ OF THE OBJECTIVE FUNCTION EQ. (1). NUMBERS REFER TO SCENARIO 4 OF TABLE IV.

	GreedyHop			**GreedyAirtime**			**GreedyMax**		
	scalar	visual	total	scalar	visual	total	scalar	visual	total
δ=0.01	9.15	12.6	21.75	31.72	8.09	39.81	22.72	10.34	33.06
δ=0.05	9.15	9.16	18.31	6.5	0.94	7.44	9.25	9.17	18.42
δ=0.1	8.31	9.16	17.47	4.27	0.93	5.2	8.31	9.16	17.47

TABLE VI
NUMBER OF ACTIVATED SENSOR NODES WHEN VARYING THE COST PARAMETER δ OF THE OBJECTIVE FUNCTION EQ. (1). NUMBERS REFER TO SCENARIO 4 OF TABLE IV

	GreedyHop			**GreedyAirtime**			**GreedyMax**		
	scalar	visual	total	scalar	visual	total	scalar	visual	total
δ=0.01	41.87	44.93	86.8	80.67	79.46	160.13	64.68	65.41	130.09
δ=0.05	41.87	44.93	86.8	30.28	28.3	58.58	42.28	45.16	87.44
δ=0.1	39.5	43.66	83.16	20.79	20.97	41.76	39.5	43.66	83.16

V. RELATED WORKS

The efficient design of general purpose, easy-reconfigurable wireless sensor network is recently attracting considerable attention in the research community. Research efforts have been put in place to realize such vision at different levels ranging from the proposal of novel programming abstractions at the sensor node level [11], [12], to the design of network-wide management platforms to orchestrate different applications on a shared physical infrastructure [13], [14].

In our previous work [3], which sets the basis for the present work, we formalize the resource allocation problem in virtual sensor network as MILP problem and we obtain the optimal solution for small-scale network instances by resorting to commercial solvers. Differently, the present work focuses on sub-optimal yet efficient greedy algorithms to solve the problem in large-scale networks.

Greedy algorithms have been applied to solve several optimization problems in telecommunications. For instance, in [15], it is applied for the optimization of the spectrum efficiency of an OFDM transmission system aiming to minimize the total allocated bandwidth considering a certain transmission data rate to each user, under the constraint of a total transmission power.

Greedy algorithms appear in network routing as well. For example, the authors of [16] introduced a greedy algorithm for Multi-Path routing in WSNs which allow to produce multiple paths between source and destination nodes in quite efficient time.

Regarding resource allocation, we can find this problem presented in [17] where a greedy-Knapsack algorithm for downlink resource allocation in LTE networks have been discussed. Also in [18], greedy algorithm was proposed for physical resource block allocation in multi-carrier wireless communications systems.

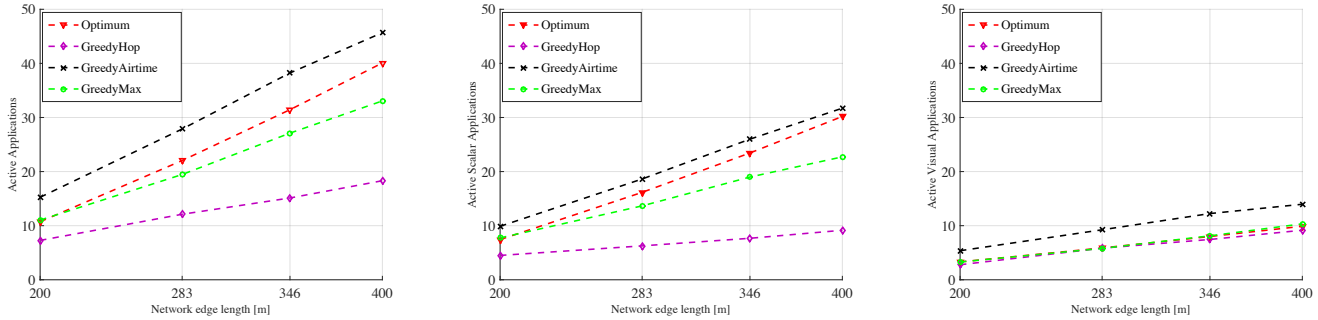
Greedy algorithms are convenient for optimization problems because they are efficient in time and often give good approximations to the optimum. And this is a common result obtained in the aforementioned works. Resource allocation problem was treated in WSNs optimization but to our knowledge this will be the first case where a greedy based algorithm is applied to resource allocation problem in virtual sensor network.

VI. CONCLUSION

We addressed in this work the problem of optimally allocate physical network resources to applications in VSNs. We proposed a greedy algorithm which maximizes the total profit involved in the deployment of multiple applications in the VSN while accounting for the constraints imposed by the physical network infrastructure. The proposed solution has been validated via numerical analysis in terms of computation time and optimality gap with respect to the optimal solution obtained with heavy, non-scalable MILP approaches. The results demonstrate that the proposed approach is simple yet effective in sensibly reducing the processing time while keeping the optimality gap below 5% in all the scenarios at hand.

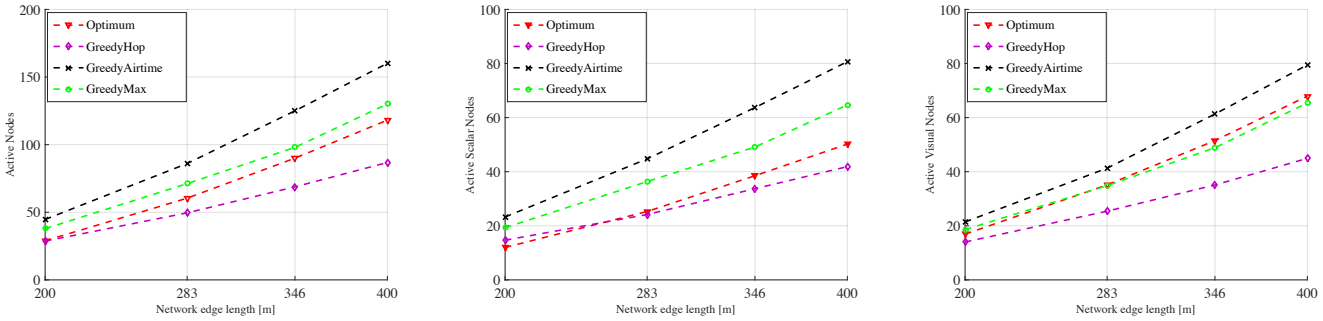
ACKNOWLEDGEMENTS

This work has been funded by the European Commission under the Erasmus Mundus GreenIT project (GreenIT for the benefit of civil society. 3772227-1-2012-ES-ERA MUNDUS-EMA21; Grant Agreement n 2012-2625/001-001-EMA2), the Spanish Government through the grant TEC2014-52969-R from the Ministerio de Ciencia e Innovación (MICINN), Gobierno de Aragón (research group T98), the European Social Fund (ESF), Universidad de Zaragoza, Fundación Bancaria Ibercaja and Fundación CAI (IT 2/15).



(a) (b) (c)

Fig. 2. Number of deployed applications. a) Total b) Scalar c) Visual



(a) (b) (c)

Fig. 3. Number of active nodes. a) Total b) Scalar c) Visual

REFERENCES

[1] L. Sarakis, T. Zahariadis, H.-C. Leligou, and M. Dohler, "A framework for service provisioning in virtual sensor networks," *EURASIP Journal on Wireless Communications and Networking*, 2012.

[2] S. Madria, V. Kumar, and R. Dalvi, "Sensor cloud: A cloud of virtual sensors," *IEEE Software*, vol. 31, no. 2, pp. 70 – 77, mar.-apr. 2014.

[3] C. Delgado, J. R. Gllego, M. Canales, J. Ortn, S. Bousnina, and M. Cesana, "On optimal resource allocation in virtual sensor networks," *Ad Hoc Networks*, vol. 50, pp. 23 – 40, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870516301007

[4] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550 (Proposed Standard), Internet Engineering Task Force, Mar. 2012.

[5] *TelosB Mote Platform Datasheet*, MEMSIC Inc.

[6] G. Coley, *Beaglebone rev a6 system reference manual*, 2012.

[7] S. Bhattacharya, A. Saifullah, C. Lu, and G. Roman, "Multi-application deployment in shared sensor networks based on quality of monitoring," in *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2010 16th IEEE*, April 2010, pp. 259–268.

[8] J. B. Javier Molina, Javier M. Mora-merchan and C. Leon, *Wireless Sensor Networks: Application*. InTech, 2010, ch. Multimedia Data Processing and Delivery in Wireless Sensor Networks.

[9] A. Redondi, M. Tagliasacchi, and M. Cesana, "Rate-accuracy optimization in visual wireless sensor networks," in *IEEE International Conference on Image Processing (ICIP2012)*, Orlando, Florida, Oct. 2012, pp. 1105–1108.

[10] A. Redondi, L. Baroffio, L. Bianchi, M. Cesana, and M. Tagliasacchi, "Compress-then-analyze vs analyze-then-compress: what is best in visual sensor networks?" *IEEE Transactions on Mobile Computing*, vol. PP, no. 99, pp. 1–1, 2016.

[11] P. Levis and D. Culler, "Mate: A tiny virtual machine for sensor networks," *SIGARCH Comput. Archit. News*, vol. 30, no. 5, pp. 85–95, Oct. 2002. [Online]. Available: http://doi.acm.org/10.1145/635506.605407

[12] J. Koshy and R. Pandey, "Vmstar: Synthesizing scalable runtime environments for sensor networks," in *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, ser. SenSys '05. New York, NY, USA: ACM, 2005, pp. 243–254. [Online]. Available: http://doi.acm.org/10.1145/1098918.1098945

[13] Y. Yu, L. J. Rittle, V. Bhandari, and J. B. LeBrun, "Supporting concurrent applications in wireless sensor networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, ser. SenSys '06. New York, NY, USA: ACM, 2006, pp. 139–152. [Online]. Available: http://doi.acm.org/10.1145/1182807.1182822

[14] I. Khan, F. Belqasmi, R. Glitho, N. Crespi, M. Morrow, and P. Polakos, "Wireless sensor network virtualization: Early architecture and research perspectives," *IEEE Network*, vol. 29, pp. 104 – 112, may-jun. 2015.

[15] J. Farah and F. Marx, "Greedy algorithms for spectrum management in OFDM cognitive systems - applications to video streaming and wireless sensor networks," November 2008.

[16] S. Masoudi, A. Rahmani, A. N. Eghbali, and A. Khademzadeh, "Gmpr: A greedy multi-path routing algorithm for wireless sensor networks," in *2008 Second International Conference on Future Generation Communication and Networking*, vol. 1, Dec 2008, pp. 25–30.

[17] N. Ferdosian, M. Othman, B. M. Ali, and K. Y. Lun, "Greedy—knapsack algorithm for optimal downlink resource allocation in lte networks," *Wirel. Netw.*, vol. 22, no. 5, pp. 1427–1440, Jul. 2016. [Online]. Available: http://dx.doi.org/10.1007/s11276-015-1042-9

[18] O. Nwamadi, X. Zhu, and A. K. Nandi, "Multi-criteria ranking based greedy algorithm for physical resource block allocation in multi-carrier wireless communication systems," *Signal Process.*, vol. 92, no. 11, pp. 2706–2717, Nov. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.sigpro.2012.04.020