Optimization of data harvesters deployment in an urban areas for an emergency scenario

Tarek Bouall, El-Hassane Aglzim, Sidi-Mohammed Senouci

Optimization of Data Harvesters Deployment in an Urban Areas for an Emergency Scenario

Tarek BOUALI, El-Hassane AGLZIM, Sidi-Mohammed SENOUCI
DRIVE Labs, University of Burgundy, Nevers, France
Email:{Tarek.Bouali,Sidi-Mohammed.Senouci,el-hassane.aglzim}@u-bourgogne.fr

***Abstract*—Since its appearance in the VANETs research community, data collection where vehicles have to explore an area and collect various local data, brings various issues and challenges. Some architectures were proposed to meet data collection requirements. They can be classified into two categories: Decentralized and Centralized self-organizing where different components and techniques are used depending on the application type. In this paper, we treat time-constrained applications in the context of search and rescue missions. For this reason, we propose a centralized architecture where a central unit plans and manages a set of vehicles namely harvesters to get a clear overview about an affected area. But, choosing the optimal number of harvesters to be deployed and the corresponding area to explore for such time-constrained applications are a real issue. In this paper, we model the problem with its constraints, then we propose a heuristic algorithm called Variable Neighborhood Search (VNS) to get the optimal number of harvesters and define areas to be explored by each one. The proposed solution combines two algorithms: The first is a greedy Best Insertion heuristic reshaped to meet our problem definition to get an initial solution and the second is a 2-Opt merged with a String Exchange heuristics which defines neighborhoods and responsible for local search and global optimization of the initial solution. Finally, the solution is analyzed regarding its optimality and the CPU calculation cost.**

***Index Terms*—VANET - Data Collection - Optimization - Harvesters - Emergency - Search and Rescue**

I. Introduction

Vehicular Ad hoc Network (VANET) is a new field which has witnessed an unprecedented growth in the research community. Thanks to the increasing evolution of embedded technologies and communication techniques, a vehicle which is the elementary component of a vehicular network is being a mobile agent equipped with a great number of sensors able to collect various kinds of data. This has impacted a huge number of human life domains from which human safety is the most important field. Assistance in case of search and rescue missions in urban areas seems to be more simple and sophisticated using vehicles' capabilities. Vehicles, while traveling through an affected area, collect many kinds of local information like videos, temperature or chemical toxicity which give a global vision about what really happening in an affected area and help to locate survivors and plan for rescue missions. Collected data by each vehicle should be delivered to a third-party to be used for continuous monitoring of the disaster area. This has led to the appearance of a new challenge namely Data Collection. This concept has snatched some interest in the VANET research community and some architectures and communication standards are proposed in the field. These architectures depend on the application constraints and the collection can be either decentralized where vehicles are autonomously organized or centralized where a static fixed third-party is responsible for the network organization.

The most investigated type of data collection architectures is decentralized, but in case of emergency scenarios where connectivity and communication between vehicles is not guaranteed, these solutions have proven inefficiency. This limitation could not be ignored especially in our work where we are dealing with the problem of emergency cars management in case of search and rescue mission in disaster urban areas. Therefore, we choose to consider a centralized solution where a central unit plans and continuously monitors a set of vehicles namely harvesters to get a global overview of the affected area. A harvester is traveling through an area defined by the central unit to collect data from vehicles and send them to the unit to be treated. However, our proposed solution brings a new challenge and two questions should be answered: How many harvester we have to deploy in a geographic area? and which area to be monitored by one harvester while considering time constraints of the application? For this reason, optimizing the number of deployed harvesters in an urban area will be the target of our actual work. Typically, many works have dealt with optimization problems but none of them has treated the problem of data harvesters deployment. So, as far as we know, we are the first to deal with this problem. We first give in this paper an analytical model that formalizes our problem, then we propose a heuristic-based solution. We have chosen to use heuristics and not an exact resolution since we are in an emergency case and the main concern is to provide a solution as soon as possible even if this latter is not optimal. The remainder of this paper is structured as follows: Section II gives an overview about data collection architectures for VANETs and some optimization studies treating either collection architectures or similar problems. Section III firstly presents an analytic model for our problem, then details the optimization algorithms we have defined. Section IV gives a case study to evaluate the performance evaluation of the algorithms. Section V concludes this paper.

II. Related Work

Since the emergence of data collection concept, many architectures were defined to get useful data from vehicles and reuse them depending on the application. In fact, these architectures can be categorized into two classes: a self-

organizing decentralized or a centralized architecture which we expose in details in the first subsection. In both classes, data is periodically collected by regular vehicles from the roads where they pass and hence give only a local overview of the monitored area. However, in a case of a disaster we need to have a global overview of the affected area. So, deploying a number of vehicles namely harvesters that are able to explore the entire monitored area and collect data, unlike the previous proposals, is a promising solution that guarantees a clear and global overview about what happens in such an area and facilitates later decisions by a data center. But, there is a challenge to be faced corresponding to how to deploy and manage data harvesters by a central third party. In the second subsection, we give an overview about the harvesters deployment problem.

A. Data Collection architectures for VANETs

A car can be defined, nowadays, as a set of non limited capabilities sensors that are able to detect and treat a huge variety of data. The collected data is exchanged between cars at each encounter while moving in a road topology using the newly standardized technique IEEE802.11p or cellular capabilities such as UMTS or LTE. Data collection is one of the most emerging treated fields in VANET community. Many architectures have been defined to support data exchange between vehicles. One of the most popular solutions are based on self-organizing architectures which can be either decentralized organization where the network is autonomous and vehicles are organizing themselves without using any external infrastructure or centralized where the vehicles organization is delegated to a third-party fixed infrastructure (e.g. RSU, eNodeB). Typically, in the case of vehicular networks, existing works are based on decentralized clustering and the election of a cluster head that collects data from cluster members to deliver it after that to its neighbors using dissemination, a routing protocol or the Store and Forward (SNF) mechanism in case of disconnections. However, in our case we are investigating the centralized solution where a central infrastructure is responsible of the planification and organization of cluster heads because we are treating an emergency case where connectivity between vehicles in an affected area is not guaranteed and the self-organization of a cluster to deliver data to a center is very difficult. For this reason, we focus on works that are based on centralized organization.

In [10] authors introduce the use of LTE eNodeB to manage the clustering in a road topology. They assume that each vehicle is equipped with both 802.11p and LTE capabilities. The 802.11p is used for the communication and data exchange between vehicles and LTE is used to interface with the eNodeB manager. In the proposed framework named LTE4V2X, each eNodeB manages the vehicles in its range by initially giving to each vehicle an ID, then it organizes the nodes into clusters and designates a cluster head. The eNodeB is responsible of the periodic update of its clusters in case of an incoming or a living node. Data is periodically collected by each cluster head, aggregated and sent to the responsible eNodeB. In [11], the same authors augment the LTE4V2X with the dissemination capabilities where they treat the case of tunnels where the eNodeB coverage is limited. They use the multi-hop forwarding between cluster heads to send data. So, the cluster head in the tunnel tries to send data by using the nearest cluster head to the eNodeB as a relay to send aggregated data.

In MobEyes [7], the authors treat the problem of data gathering about terrorist attacks and crimes. They aim to build a cooperation between civils and police agents to collect data that can be useful earlier to identify outlaw peoples and rebuild crime scenes. They consider all vehicles (police agents and civil cars) are equipped with cameras and chemical sensors and able to collect all activities. For this reason, a framework was built and designed to work depending on the type of the car. Civil cars collect a huge amount of data, store it locally and periodically generate a set of summaries to be later diffused in the network in each encounter with other cars. Agent vehicles are responsible of data harvesting from the civil cars to deliver it to the agency. For this reason the designed MobEyes works in a different way. In fact, there were two types of functioning: reactive harvesting and proactive harvesting. In reactive harvesting, agents are mobilized when an abnormal activity is detected to collect data from civil vehicles by identifying missing data using a bloom filter and periodically diffusing the set of summaries they have to obtain in response missing data. In proactive harvesting, police agents are always mobile to collect data and build a low-cost distributed index which contains all harvested data.

As said above, because we are in an emergency use case, we assume that we have only a data center that makes decisions and manages emergency vehicles from the beginning of their mobilization to an affected area. For this reason, we consider that the management of data harvesters should be centralized and delegated to that center which schedules the movement plans and the areas to be visited.

B. Optimization of Data Harvesters Deployment

Optimizing harvesters in an urban area is a nested problem that contains two main fields to be treated which are the optimization of each harvester movement circuit and the optimal number of these harvesters. As far as we know, we are the first to treat this problem in VANET. However, many works were proposed treating location optimization problems and the most known are facilities location, covering and Vehicle Routing Problem (VRP). In facilities location problem [2], each facility should be optimally placed in order to distribute goods to clients at a minimum cost. [8] treats a problem of locating the charging stations to serve electricity demands of electric vehicles. A Genetic algorithm is used to minimize the investment and transportation costs.

The covering location problem is how to find optimal positions to cover all clients demands (e.g RSU placement). In [1] a Road Side Unit placement scheme is optimized. The main used criteria is the delivery time of an abnormal activity

detected by a vehicle in a road segment to the nearest RSU. In this work, the authors consider intersections as potential candidates where to place RSUs and try to choose between these locations to cover the maximum area and minimize the delivery time. They compare two heuristics: Balloon Expansion Heuristic (BEH) and Binary Integer Programming (BIP) to solve the problem.

In VRP the problem is how to manage a set of vehicles to optimally distribute goods to a number of dispersed clients while respecting capacity constraints of vehicles, the priority of demands and time intervals of delivering. In [5], authors treat the problem of emergency vehicles planning to serve emergency calls. They consider a number of stations where each one is delegated to manage a set of vehicles and satisfy calls in a predefined area. In this work, an Ant-Tabu algorithm is used to minimize a number of functions (e.g. time to a station, time to a call, etc.). In [3] authors compare exact algorithms, heuristics and meta-heuristics to solve the VRP with time window constraints. One of the used heuristics are the construction and amelioration where they used a greedy Best Insertion algorithm to build an initial solution and a Variable Neighborhood Search (VNS) heuristic [4] for optimization. In the VNS a combination of six heuristics were defined as neighborhood relations. Three mono-tour heuristics: 2-Opt, Or-Opt, 2-Exchange and three multi-tours: String Relocation, String Exchange and String Cross.

Most of the above treated problems and proposed solutions and especially the VRP problem seem to be similar to ours but differ in the way that their objective is minimization. However, in our case we aim to maximize the distance covered by each harvester while considering time constraints. We should, also, take into account that each road segment is covered unlike the VRP where the objective is to visit some points without considering roads.

III. Data Harvesters Deployment Formulation

In our problem, we aim to maximize the covered distance by each harvester in an urban area without violating the time constraint between two successive visits for the same region. So, the global problem of harvesters optimization is divided into a set of sub-problems of tours optimization. In this section, we first give the mathematic model to formulate the problem and after that a solution is detailed.

A. Problem Modeling

For the sake of clarity, lets introduce our proposed architecture for data collection in an emergency scenario. We aim to deploy a number of vehicles called harvesters where each of them is delegated to a specified area and has a specific circuit to explore and harvest data from vehicles it meets and send them to a data center using a specific routing protocol. In this work, we target to define harvesters circuits and areas to explore. The idea is to divide the roads topology into segments separated by intersections. So that the urban area is modulated as a graph where intersections are vertices and road segments are edges and a harvester should travel a portion of the graph and visit a number of edges. Typically, we should define an objective function to be optimized and a number of constraints to be respected, but before that lets introduce the set of parameters used in this section and represented in Table I:

Parameter	Definition
V	Set of all vertices/intersections: i$\in$ V
E	Set of edges/road segments: (i,j)$\in$ E
z_{ij}	=1 if segment (i,j) exists else =0
l_{ij}	length of segment (i,j)
t_{ij}	Time to travel a segment (i,j)
v_{ij}	Maximum speed of a harvester in a segment (i,j)
y_{ij}^k	=1 if the segment (i,j) is visited by a harvester k else =0
T_{MAX}	The maximum time between two successive visits to a point in the segment

TABLE I: Notations

To minimize the number of harvesters, we should maximize the portion of the graph covered by a harvester which means the maximization of the traveled distance. So the problem is divided into k sub-problems. Let F be the distance function to be maximized by a harvester k where n is the number of vertices (intersections):

$$F = Max \sum_{i=0}^{n} \sum_{j=0}^{n} l_{ij} y_{ij}^k \tag{1}$$

The feasibility of the solution depends on different constraints represented by the following equations: Eq.2 guarantees that the maximum time between two successive visits to a point in a segment should not exceed a threshold T_{MAX} defined by the application, Eq.3 ensures that every road segment should be visited at least once by all the harvesters where m is the number of harvesters and Eq.4 is the integrity constraint.

$$\sum_{i=0}^{n} \sum_{j=0}^{n} t_{ij} y_{ij}^k \leq T_{max} \tag{2}$$

$$\sum_{k=0}^{m-1} y_{ij}^k \geq 1 \tag{3}$$

$$y_{ij} \in \{0,1\} \tag{4}$$

In the following, we present a solution for the problem by defining a number of heuristics we use.

B. Optimization Solution

As cited above, we are dealing with an emergency scenario where time is a rigid constraint. Hence, the algorithm to be used to solve the harvesters' deployment problem should give results in a reasonable time. For this reason, we opt for meta-heuristics. It is clear that they don't give always the optimum solution but they approximately hit optimality in a reasonable time. To deal with our problem, we modified a famous meta-heuristic which is the Variable Neighborhood Search [4]. VNS is a meta-heuristic which always starts with an initial solution obtained by a construction heuristics, then it defines a number of operators N_k (k=1..k_{max}) to be

applied to this state (current solution) resulting in a set of successor states. The set of obtained states by applying an operator is called a neighborhood and it is optimized by local search techniques. For the construction of initial solution, we reshape the greedy Best Insertion (BI) heuristic to respect our specification and constraints. However for neighborhoods, we have defined an heuristic which is the combination of two well known heuristics: 2-Opt used for TSP [6] which is a Local Search Heuristic and String Exchange [9] which aims to globally optimize the solution by exchanging k strings between two circuits (tours). Figure 1 introduces the global functioning and transitions in the VNS. More details are presented below.

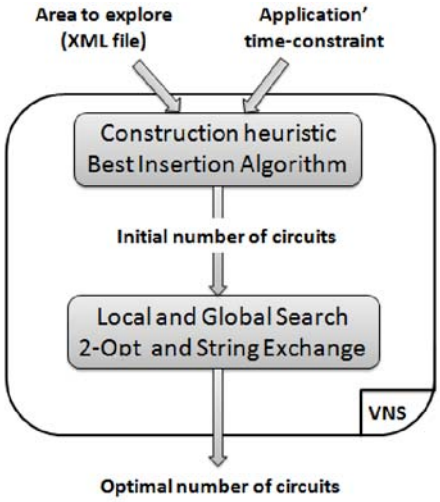

Fig. 1: VNS Functioning

1) Construction heuristic: In this section we detail the Best Insertion heuristic modifications. Traditionally, BI is used for minimization problems and the construction deals always with a set of points (graph vertices) which are mostly cities in case of TSP or clients in case of VRP. However in our case, the objective is maximization and we have to deal with road segments (graph edges). So we modify the BI to accept edges while building a cycle and consider the maximum length as the adding criteria without forgetting the time constraint of the application. The algorithm is detailed below:

Algorithm 1: Modified BI Algorithm

Input: A valuated graph with (d_{ij},t_{ij}) as cost of segment (i,j)
Output: A set of circuits delimiting the areas covered by each harvester
1- Choose a random segment as first cycle
2- Browse all segments and add the one with maximum length
3- Repeat 2 until $\sum_{i=0}^{n}\sum_{j=0}^{n} t_{ij}y_{ij}^{k} \geq T_{max}$
4- if $\sum_{i=0}^{n}\sum_{j=0}^{n} t_{ij}y_{ij}^{k} \geq T_{max}$ then
choose the last segment as the first cycle of the new circuit
5- Repeat 2,3 and 4
6- Stop the algorithm when there is no more segments to add

This algorithm results in a set of circuits. These circuits are later used in the optimization algorithm to be improved to hit the optimal solution that gives the number of harvesters to be deployed.

2) Neighborhood heuristic: In this section we take the initial solution given by the Best Insertion heuristic and we make amelioration at each iteration to finally give an optimal solution. For this reason, we have defined an heuristic combining the 2-Opt and String Exchange. This heuristic defines the VNS neighborhood used for local and global optimum search. In the algorithm, we take the set of circuits produced by the initial solution, we browse all of them and choose at each iteration two adjacent circuits. We firstly browse local segments of each circuit and try to exchange tow segments if this action ameliorates the time consumption. Then, we compare the two circuits and see if we can exchange segments between them. The exchange is made if it minimizes the number of travels for one segment or minimizes the time of the circuit. The different instructions of the amelioration algorithm are detailed below with N is the number of circuits produced by BI algorithm, C_i the circuit i, K^i the number of segments in C_i, S_j^i is the segment j in the circuit i.

Algorithm 2: Amelioration Algorithm

Input: -Set of harvesters circuits produced by the BI algorithm
-The distance of each circuit
-Time to travel each Circuit
Output: A set of optimal routes (circuits) delimiting the areas covered by each harvester
1- for i=0..N
2- Choose C_i and C_{i+1}
3- For j=0..K^i-1
4- For k=j+1..K^i
5- Make a virtual exchange between the two segments S_j^i and S_k^i
6- Recalculate the new configuration time and compare it with the initial time
7- If the new time is less than the initial one then validate exchange
8- For h=0..$K^{i+1}-1$
9- For x=h+1..K^{i+1}
6- Do 5, 6 and 7 with S_h^{i+1} and S_x^{i+1}
7- Make a virtual exchange between S_j^i and S_h^{i+1}
8- Recalculate the two times of C_i and C_{i+1}
9- If there is an amelioration of time, then validate the exchange

The heuristic we have defined uses a local search approach given by the 2-Opt and a global optimization by comparing adjacent circuits to make improvements which give a global optimum for the problem. The algorithm gives a number of optimal circuits as final result. In our case, and as said above, a harvester is delegated to explore/monitor an area and travel a defined number of road segments (circuit) which means that the number of harvesters will be the same as the circuit number.

IV. Experimental Results

In this section, we evaluate our proposed solution using the Manhattan road topology and we expose results to discuss the ability of our heuristic regarding its optimality and its time calculation cost. Manhattan is one of the most dense borough of the New York city, it is of $87.46Km^2$ of size with approximately $28Km^2$ the size of water. Most of the Manhattan roads are unidirectional which leads sometimes to a greater time consumption to reach a near point and can affects the final results of the proposed algorithm.

We firstly, evaluate the number of harvesters given by the initial solution (Best Insertion algorithm) and the final one given by the neighborhood heuristic. Then, we calculate the CPU time consumed by each of them to give the solution. Initially, we set T_{MAX} to one hour and we start the execution. Results are summarized in table II where each value represents an average of twenty runs.

We can observe that the number of harvesters given by the initial solution is approximately optimal and the time consumed by the Best Insertion algorithm is higher than the Neighborhood time. This is due to the higher value of T_{MAX} which gives the possibility to the BI to loop several times and increase the circuit length as the application' time-constraint is respected. So, in this case, we can limit our optimization to the BI and we don't need to consume more time using the neighborhood heuristic.

Algorithm	**Harvesters' Number**	**CPU Time (ms)**
Best Insertion	24	47
Neighborhood	21	30

TABLE II: Best Insertion vs Neighborhood

In Fig.2, we depict the evolution of the number of harvesters depending on the application' time-constraint. The x axis represents the T_{MAX} value increasing from 15min to 2hours and y axis represents the number of deployed harvesters for each value of T_{MAX}. The figure shows the impact of the time-constraint rigidity on the final solution of the algorithm where the number of harvesters rapidly decreases when T_{MAX} increases. We need a high number of harvesters for T_{MAX}=15min (73 harvesters) where we need only 5 harvesters for T_{MAX}=2hours considering, in this first case, that we should collect information and take a global overview of a topology of $87Km^2$ in 15 min.

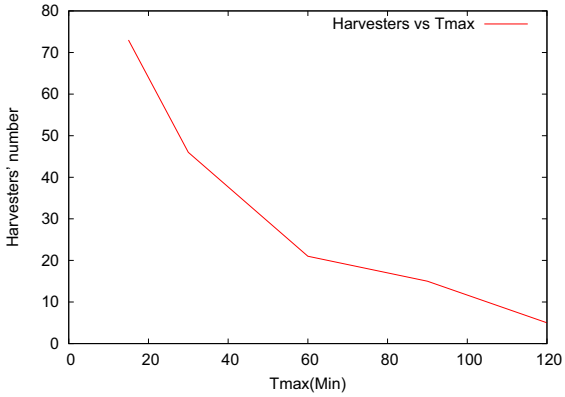

Fig. 2: The number of harvesters vs Tmax

Fig.3 highlights the average CPU calculation time when the time-constraint increases. We run the algorithms 10 times for each value of T_{MAX} and we calculate the average time of the CPU calculation. We notice that the calculation time of the proposed algorithms increases linearly from 15ms for T_{MAX}=15min to 81ms for T_{MAX}=2hours. The time increasing is due to the number of loops executed by each algorithm because the time-constraint is being more and more higher which means that it won't be exceeded rapidly and the search is larger. However, it stays always reasonable and doesn't take exponential values.

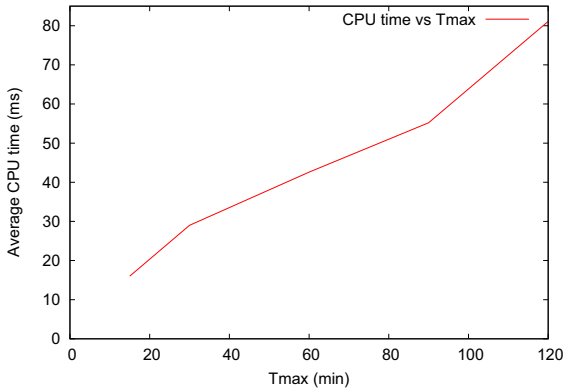

Fig. 3: The average of CPU calculation time vs Tmax

As we know, in a case of a chemical attack or fire, the size of the affected area rapidly increases to affect the surroundings. So, the given algorithms should continuously give faster solutions regarding the size of the area. To evaluate the ability of our algorithms to give an optimal solution at a minimum calculation time cost, we applied them to different sizes of the Manhattan topology. We consider different percentages of the total topology as affected areas increasing from 15 to 100%. Results, in Fig.4, shows that the CPU time of our proposed algorithms increases linearly when the size of the affected area increases. This means that even for a very large topology, the time to calculate the optimal solution stays reasonable and doesn't jump rapidly to an exponential value.

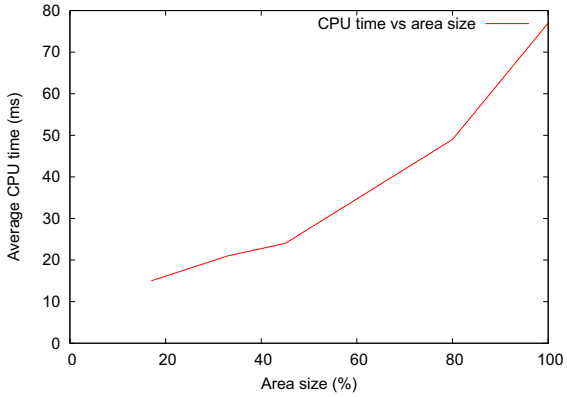

Fig. 4: The CPU calculation time vs topology size

Regarding the given results in Table II, Fig.2, 3 and 4 we notice that our algorithm is able to deliver an optimal solution at a very reasonable time even if the size of the topology to be monitored is very large or the time-constraint increases. These results prove the higher performance of the given algorithms. In Table II, we also conclude that the Best Insertion is able to

give an approximative optimal solution especially for a high value of T_{MAX} which means that we can minimize the CPU time by limiting calculation to the first initial solution.

V. Conclusion

Data collection is a very challenging field in the VANET research community. This concept defines how data is being retrieved from vehicles in a geographic area to a third-party of interest. Many self-organizing centralized or decentralized architectures were defined to support data collection. In this work, we also propose a new centralized architecture where a central unit monitors a number of vehicles called harvesters. However, we should firstly define a minimum number of harvesters to be deployed without violating time constraints imposed by the application. Deployed harvesters should give a clear overview about an affected area in a rescue mission. For this reason, we first analytically model the problem to identify objective functions, then we propose a Variable Neighborhood Search (VNS) heuristic to solve the problem. We modify the greedy Best Insertion heuristic to give an initial solution for the problem. The given solution is used as an input for the neighborhood heuristic defined by the combination of the 2-Opt and String Exchange heuristics to get a final global solution. The adoption of heuristics and not exact resolutions is imposed by the emergency scenario we are dealing with. Experimental results given by the application of the algorithms to a Manhattan topology prove the efficiency of our proposals regarding the processing time and the number of resulting harvesters. As future work, we aim to compare our work to exact solutions and we also plan to investigate on social routing to route data from a harvester to the central unit.

Acknowledgment

This work has been funded by the Europeen project Car-Code.

References

[1] A. Baber, F. Amjad, and C. Zou. Optimal roadside units placement in urban areas for vehicular networks. *2012 IEEE Symposium on Computers and Communications (ISCC)*, 0:000423–000429, 2012.

[2] M-A. El-Baz. A genetic algorithm for facility layout problems of different manufacturing environments. *Comput. Ind. Eng.*, 47(2-3):233–246, November 2004.

[3] É. Grellier. *Optimization of vehicle routing in the context of reverse logistics: modeling and resolution using hybrid methods*. 2008.

[4] P. Hansen and N. MladenoviÄĞ. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449 – 467, 2001.

[5] S. Ibri, H. Drias, and M. Nourelfath. A parallel hybrid ant-tabu algorithm for integrated emergency vehicle dispatching and covering problem. *Int. J. Innov. Comput. Appl.*, 2(4):226–236, November 2010.

[6] D.S. Johnson and L. Mcgeoch. *The Traveling Salesman Problem: A Case Study in Local Optimization*, pages 1–103. 1997.

[7] U. Lee, B. Zhou, M. Gerla, P. Bellavista, A. Corradi, and E. Magistretti. Mobeyes: smart mobs for urban monitoring with a vehicular sensor network. *Wireless Commun.*, 13(5):52–57, October 2006.

[8] S. Mehar and S-M. Senouci. An optimization location scheme for electric charging stations. *IEEE SaCoNet*, pages 1–5, 2013.

[9] P.C. Pop, C.P. Sitar, I. Zelina, V. Lupse, and C. Chira. Heuristic algorithms for solving the generalized vehicle routing problem. *International Journal of Computers Communications & Control*, 6(1):158–165, 2011.

[10] G. Remy, S-M. Senouci, F. Jan, and Y. Gourhant. Lte4v2x: Lte for a centralized vanet organization. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–6, 2011.

[11] G. Remy, S-M. Senouci, F. Jan, and Y. Gourhant. Lte4v2x - collection, dissemination and multi-hop forwarding. In *ICC*, pages 120–125. IEEE, 2012.