On Different-Dimensional Deployment Problems of Hybrid VANET-Sensor Networks with QoS Considerations

Chun-Cheng Lin* · **Peng-Chung Chen** · **Li-Wei Chang**

Abstract To enhance vehicular ad hoc networks (VANETs) with the ability to detect road conditions, hybrid VANET-sensor networks (HVSNs) additionally deploy a large number of wireless sensors on road sides, so as to provide a safer driving environment. Most of the previous works on network deployment problems considered too simplified problem settings, and few of them focused on deploying HVSNs. To meet more practical requirements, this paper investigates the problem of deploying roadside units (RSUs) to serve sensors on three different-dimensional road settings (i.e., a one-dimensional road, a road represented as grid points, and a dense network environment) in HVSNs with two objectives (i.e., to minimize the total distance and the total number of hops from sensors to their respective closest RSUs) and two QoS constraints (i.e., for maximal number of hops and capacity). Achieving the two objectives can effectively decrease the probability of transmission interference and lead to shorter transmission time to inform drivers of unexpected events. Since the optimal deployment problems for a grid road and a dense HVSN cannot be solved analytically, this paper creates mathematical programming models for the problems, and further proposes an improved harmony search algorithm. By simulation in different-size instances and statistical analysis, the proposed approaches show promising performance in solving the problems.

Keywords Vehicular network · wireless sensor network · deployment · QoS · metaheuristic · harmony search algorithm

1. Introduction

To increase safety of road use and convenience of transmitting emergent traffic messages, vehicular

C.-C. Lin · P.-C. Chen · L.-W. Chang
Department of Industrial Engineering and Management,
National Chiao Tung University, Hsinchu 300, Taiwan
e-mails: cclin321@nctu.edu.tw, skybird810604@gmail.com, chang.leewei@msa.hinet.net

* C.-C. Lin is the corresponding author of this paper (E-mail: cclin321@nctu.edu.tw).

ad-hoc network (VANET) integrates wireless networks and roadside units (RSUs) to communicate with users in vehicles via vehicle-to-RSU and vehicle-to-vehicle transmissions. A good deployment of RSUs in VANETs can promote driving safety remarkably, e.g., by deploying RSUs on the locations with heavy traffic flow or high accident frequency. Hence, deploying RSUs in VANETs has been studied increasingly. For instance, the work in [1] considered deployment of multiple RSUs to cover a number of access points (APs, serving as relay nodes between users in vehicles and RSUs) with given locations on a one-dimensional road, a two-dimensional dense network, and a grid network, so that the total number of hops from APs to their closest RSUs is minimized. Later, the work in [2] extended the work to the case of multiple-lane roads with a cost-effective strategy. Different methods for optimize placement of RSUs were also proposed, e.g., genetic algorithm [3], [4], randomized algorithm [5], and dynamic method [6].

However, VANET may not always guarantee to detect road hazards (e.g., a car accident outside the transmission range) and special road conditions (e.g., frozen or waterlogged roads) in time owing to high traffic flow, unpredictability, and deployment of too few RSUs. Additionally, when density of vehicles on a road is too low, communications in VANETs could not be maintained. To alleviate these issues, some improvements on VANET technologies have been made, e.g., quantification of VANET access probability [7], connectivity [8], [9], and link quality estimation in VANETs [10].

On the other hand, the works in [11], [12] proposed hybrid VANET-sensor networks (HVSNs), consisting of RSUs, sensors, and moving vehicles, in which the former two types of devices are deployed on road sides to provide communication for moving vehicles. Each sensor is equipped with a ZigBee interface for communication with other sensors and vehicles. Each RSU or moving vehicle is equipped with a WiFi interface for communication with other vehicles and RSUs; and a ZigBee interface for communication with sensors. HVSNs has the merits of both VANET and sensor networks that can strengthen each other. Aside from RSUs, sensors are installed on road sides for their advantages of low power consumption and promising detection ability [13].

Later, the work in [14] investigated the optimal deployment of RSUs and sensors along the two sides of a road represented as grid points in HVSNs, which aims to minimize the total cost of installing RSUs and sensors so that RSUs and sensors are connected and can cover all grid points of the road. Then, they established a mathematical programming model for the problem, and showed the problem to be NP-complete by reduction from an NP-complete 2D critical grid coverage problem [15]. Recently, our previous work in [16] has extended [14] to a two-lane load case in which RSUs and sensors allow to be

deployed on the two sides and the middle island of the road, and further proposed a center particle swarm optimization approach.

This paper extends our previous work in [16] to multi-dimensional roads and takes QoS into consideration to make this work meet more practical requirements. The main motivations behind this work are explained as follows.

- Our previous work in [16] did not consider various network structures. Inspired from the work in [1] that considered deployment of vehicular networks on a one-dimensional network, a dense network, and a grid network, this paper extends our previous work in [16] with more network structures.
- To the best of our understanding, no previous works on HVSNs had QoS considerations. However, QoS has been studied widely in different networks, e.g., link stability for QoS [17], [18], QoS for VANET [19], and gateway placement with QoS [20]. In practice, QoS is very crucial in HVSNs because considering QoS can avoid missing messages and delays during transmission, which further promote traffic safety. Hence, this paper considers two QoS constraints, and the objective of the concerned problem is also related to QoS.
- Previous main related works (e.g., [1] and [2]) considered to minimize the total number of hops or installation cost. The problem concerned in this paper considers two objectives: the total distance and the total number of hops from sensors to their respective closest RSUs, in which achieving the former objective can reduce possibility of missing messages and transmission interference, and reduce the transmission time so that users can receive emergent messages efficiently and respond to them in time; and achieving the latter objective can reduce the delay time of transmission,

In real world, the information on specific locations of a road that have frequent traffic events or road conditions is known, e.g., some road locations often lead to traffic accidents or are waterlogged after a heavy rain. Hence, sensors must be installed near those specific road locations, and hence, their locations are fixed and known to us. Based on this, assuming that locations of sensors are given, this paper considers the optimal deployment of RSUs on different-dimensional road settings in HVSNs. The comparison of this work with previous main related works are listed in Table 1, among which the main differences are explained as follows. Different from the previous works, this paper proposes to minimize not only number of hops but also the total distance from sensors to RSUs on two-dimensional roads or area, and also considers two QoS constraints (i.e., for the maximal number of hops and capacity). Since the two-dimensional deployment problems are NP-complete, this paper further proposes and implements

the metaheuristic approaches based on harmony search algorithm (HSA) for those problems.

Table 1. Comparison of this work with previous related works.

Reference	Network type	Objective	Road	Middle island	Decision variables	Locations of sensors/APs	QoS	Method
[1]	VANET	Minimize number of hops	1D road	No	Location of a gateway	Fixed	No	Analytic
			1D road	No	Locations of multiple gateways	Fixed	No	Analytic
			2D dense network	No	Locations of multiple gateways	Fixed	No	Heuristic
			2D grid network	No	Locations of multiple gateways	Fixed	No	Analytic
[14]	HVSN	Minimize costs of RSUs and sensors	2D one-lane grid road	No	Locations of multiple RSUs and sensors	Not fixed	No	ILP
[16]	HVSN	Minimize costs of RSUs and sensors	2D two-lane grid road	Yes	Locations of multiple RSUs and sensors	Not fixed	No	ILP, heuristic
This work	HVSN	Minimize the total distance and the total number of hops	1D road	No	Location of an RSU	Fixed	No	Analytic
			1D road	No	Locations of multiple RSUs	Fixed	No	LP
			2D grid road	Yes	Locations of multiple RSUs	Fixed	Yes	ILP, heuristic
			2D dense network	No	Locations of multiple RSUs	Fixed	Yes	MIP, heuristic

2. The HVSN Deployment Problems for a One-Dimensional Road and Our Solutions

This section considers two simplified versions of the HVSN deployment problem for a one-dimensional road based on the work in [1], and provides solutions for the two problems. In the work in [1], one or multiple gateways are deployed to serve multiple APs on a one-dimensional road in vehicular networks so that the total number of hops is minimized. The main difference of the models in this section from the work in [1] is that our model focuses on HVSNs (i.e., to deploy RSUs to serve sensors), and changes to minimize the total distance from sensors to their respective closest RSUs, which also achieves minimization of the total number of hops.

2.1. Deploying a single RSU on a one-dimensional road

Consider a one-dimensional road shown in Fig. 1(a), in which positions of m sensors are given, and they are labeled by S_1, S_2, …, S_m from the left to the right along the road. The problem in this subsection to find a position X along the road and deploy an RSU at this position, so that the total sum of the transmission distance from each sensor to the RSU is minimized, i.e., the problem objective is as follows:

$$\text{Minimize} \quad |X - S_1| + |X - S_2| + ... + |X - S_m| \tag{1}$$

The solution of the above problem can be obtained easily as follows. When m is odd, X is the median of $S_1, S_2, \ldots, S_m$, i.e., $X = S_{(m+1)/2}$. When m is even, X can be located at any location between $S_{m/2}$ and $S_{m/2+1}$.

(a) (b)

Fig. 1. Illustration of deploying (a) a single RSU and (b) n RSUs to serve m sensors.

2.2. Deploying multiple RSUs on a one-dimensional road

Since not only one RSU is deployed on a road, this subsection considers to deploy multiple RSUs on the road, as shown in Fig. 1(b), in which n RSUs are deployed to serve m servers. Assume that $n > m$; otherwise, it is trivial to locate m RSUs at m sensor positions. Let $X_1, X_2, \ldots, X_n$ denote positions of the n RSUs along the road, which are decision variables of the problem. Then, this problem can be described as the following linear programming model, and hence can be solved in polynomial time.

$$\text{Minimize} \quad \sum_{j=1}^{m} L_j \tag{2}$$

s.t.

$$L_j = a_{1j} \mid X_1 - S_j \mid + a_{2j} \mid X_2 - S_j \mid + \ldots + a_{nj} \mid X_n - S_j \mid, \ \forall j \in \{1,2,\ldots,m\}; \tag{3}$$

$$a_{1j} + a_{2j} + \ldots + a_{nj} = 1, \ \forall j \in \{1,2,\ldots,m\} \tag{4}$$

where L_i is the distance from sensor i to its closest RSU; and a_{ij} is a binary variable: if the RSU closest to sensor i is RSU j, then $a_{ij} = 1$; otherwise, $a_{ij} = 0$.

In above model, Objective (2) is to minimize the total transmission distance, which is computed by the total of all L_i's. Each variable L_i is determined by Constraints (3) and (4). Given some j, Constraint (4) enforces only one an a_{ij} to be one for some i, so that $L_j = |X_i - S_j|$ in Constraint (3). And, since Objective (2) minimizes all positive L_j's, and $n > m$, it implies to find some X_i's so that $L_j = |X_i - S_j|$ is minimized.

3. The Two-dimensional HVSN Deployment Problem for a Road Represented as Grid Points

This section first creates a mathematical programming model for the two-dimensional HVSN deployment problem for a road represented as grid points, and then provides an HSA approach.

3.1. Problem model

Consider a larger-size multi-lane road with a middle island. Hence, RSUs can be installed on the two sides and the middle island of the road. To precisely analyze whether the whole road surface is covered, the two-dimensional road surface is represented as $(H+1)\times(W+1)$ grid points as illustrated in Fig. 2, in which positions of sensors are fixed and can only be at grid points of the two sides, i.e., the 0-th and H-th rows; and positions of RSUs need to be determined and can only be at grid points of the two sides and the middle island of the road, i.e., grid points at the 0-th, $H/2$-th, and H-th rows.

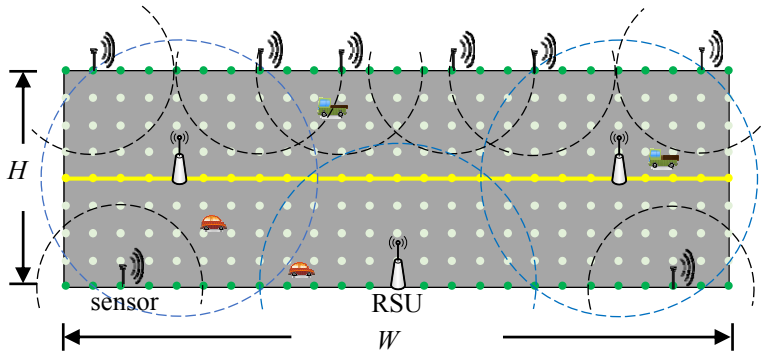

Fig. 2. Illustration of the HVSN deployment problem for a two-dimensional road represented as grid points.

Aside from the sensing function, each sensor also has the transmission function, and hence, has a transmission coverage range, which is illustrated as the plotted circle associated with each sensor in Fig. 2. Generally, the transmission range of an RSU is greater than that of a sensor. The HVSN deployment concerned in this section is to find positions of RSUs to cover all grid points of the road except for those covered by sensors, so that both the total distance and the total number of hops from sensors to their respective closest RSUs are minimized, while the following two QoS constraints must be satisfied:

- *Constraint of maximum number of hops*: This paper restricts the number of hops of each transmission from a sensor to an RSU to be at most two, i.e., there are only two cases: a sensor to an RSU directly (i.e., a sensor is covered by an RSU) or indirectly via another sensor relaying (i.e., a sensor is covered by another sensor that is covered by an RSU).

- *Capacity constraint*: In this paper, the total number of sensors covered by each RSU is restricted by an upper bound.

Since coverage of a sensor is smaller than that of an RSU, a sensor located at the middle island of the road has lower effectiveness than an RSU with larger coverage located at the middle island. Hence, this paper restricts sensors to be located at the two road sides. This problem can be modelled by an integer

nonlinear program. The decision variables in this model are as follows:

- (X_j, Y_j): Location of RSU j, which must be at the two sides and the middle island of the road (i.e., $X_j \in \{0, H/2, H\}$).

The variables used in this model are as follows:

- $a_{ij} = \begin{cases} 1, & \text{if sensor } i \text{ is covered by RSU } j; \\ 0, & \text{otherwise.} \end{cases}$
- $b_{ij} = \begin{cases} 1, & \text{if sensors } i \text{ and } k \text{ cover each other;} \\ 0, & \text{otherwise.} \end{cases}$
- $c_{uvi} = \begin{cases} 1, & \text{if grid point } (u,v) \text{ is covered by sensor } i; \\ 0, & \text{otherwise.} \end{cases}$
- $d_{uvj} = \begin{cases} 1, & \text{if grid point } (u,v) \text{ is covered by RSU } j; \\ 0, & \text{otherwise.} \end{cases}$

The parameters and indices used in this model as follows:

- j: Index of an RSU.
- i, k: Indices of sensors.
- (u, v) : Location of the grid point at the u-th row and the v-th column of the road.
- n: Number of RSUs.
- m: Number of sensors.
- C_R: Coverage of an RSU.
- C_S: Coverage of a sensor.
- D_{ij}: The distance from sensor i to RSU j.
- $L_{:i}$ The distance from sensor i to its closest RSU.
- H_i: Number of hops from sensor i to its closest RSU.

- E_{uvi}: The distance from grid point (u, v) to sensor i.
- F_{uvj}: The distance from grid point (u, v) to RSU j.
- Q: Capacity of an RSU, i.e., the maximum number of sensors that an RSU can serve (i.e., the QoS constraint for relay).
- (P_i, Q_i): Location of *sensor i*.

The mathematical programming model is detailed as follows:

$$\text{Minimize} \quad \lambda \cdot \frac{\sum_{i=1}^{m} L_i}{m \cdot (C_R + C_S)} + (1-\lambda) \cdot \frac{\sum_{i=1}^{m} H_i}{2m} \tag{5}$$

s.t.

$$D_{ij} = \sqrt{(X_j - P_i)^2 + (Y_j - Q_i)^2}\ , \ \forall i \in \{1,2,...,m\}, \forall j \in \{1,2,...,n\}\ ; \tag{6}$$

$$X_j \in \{0, H/2, H\},\ Y_j \in \{0, 1, \ldots, W\},\ \forall j \in \{1, 2, \ldots, n\}; \tag{7}$$

$$L_i = \sum_{j=1}^{n} a_{ij} \cdot D_{ij} + \sum_{k=1, k\neq i}^{m} \sum_{j=1}^{n} b_{ikj} \cdot (\sqrt{(P_i - P_k)^2 + (Q_i - Q_k)^2} + D_{kj}),\ \ \forall i \in \{1,2,...,m\}\ ; \tag{8}$$

$$\sum_{j=1}^{n} a_{ij} + \sum_{k=1}^{m} \sum_{j=1}^{n} b_{ikj} = 1\ ,\ \forall i \in \{1,2,...,m\}\ ; \tag{9}$$

$$a_{ij} \in \{0,1\}\ ,\ \forall i \in \{1,2,...,m\}, \forall j \in \{1,2,...,n\}\ ; \tag{10}$$

$$b_{ikj} \in \{0,1\}\ ,\ \forall i,k \in \{1,2,...,m\}, k \neq i, \forall j \in \{1,2,...,n\}\ ; \tag{11}$$

$$a_{ij} \cdot D_{ij} \leq C_R\ ,\ \forall i \in \{1,2,...,m\}, \forall j \in \{1,2,...,n\}\ ; \tag{12}$$

$$b_{ikj} \cdot D_{kj} \leq C_R\ ,\ \forall k \in \{1,2,...,m\}, \forall j \in \{1,2,...,n\}\ ; \tag{13}$$

$$b_{ikj} \cdot \sqrt{(P_i - P_k)^2 + (Q_i - Q_k)^2} \leq C_S\ ,\ \forall i,k \in \{1,2,...,m\}, k \neq i, \forall j \in \{1,2,...,n\}\ ; \tag{14}$$

$$H_i = \sum_{j=1}^{n} a_{ij} + \sum_{k=1}^{m} \sum_{j=1}^{n} 2b_{ikj}\ ,\ \forall i \in \{1,2,...,m\}\ ; \tag{15}$$

$$\sum_{i=1}^{m} a_{ij} \le Q \ , \ \forall j \in \{1,2,...,n\} \ ; \tag{16}$$

$$E_{uvi} = \sqrt{(u-P_i)^2 + (v-Q_i)^2} \ , \ \forall u \in \{0,1,...,H\}, \forall v \in \{0,1,...,W\}, \forall i \in \{1,2,...,m\} \ ; \tag{17}$$

$$F_{uvj} = \sqrt{(u-X_j)^2 + (v-Y_j)^2} \ , \ \forall u \in \{0,1,...,H\}, \forall v \in \{0,1,...,W\}, \forall j \in \{1,2,...,n\} \ ; \tag{18}$$

$$c_{uvi} \cdot E_{uvi} \le C_S \ , \ \forall k \in \{0,1,...,H\}, \forall t \in \{0,1,...,W\}, \forall i \in \{1,2,...,m\} \ ; \tag{19}$$

$$d_{uvj} \cdot F_{uvj} \le C_R \ , \ \forall u \in \{0,1,...,H\}, \forall v \in \{0,1,...,W\}, \forall j \in \{1,2,...,n\} \ ; \tag{20}$$

$$c_{uvi} \in \{0,1\} \ , \ d_{uvj} \in \{0,1\} \ , \ \forall u \in \{0,1,...,H\}, \forall v \in \{0,1,...,W\}, \forall i, j \in \{1,2,...,n\} \ ; \tag{21}$$

$$\sum_{i=1}^{m} c_{uvi} + \sum_{j=1}^{n} d_{uvj} \ge 1 \ , \ \forall u \in \{0,1,...,H\}, \forall v \in \{0,1,...,W\} \ . \tag{22}$$

Objective (5) is to minimize both the total distance (i.e., $\sum_{i=1}^{m} L_i$) and the total number of hops (i.e., $\sum_{i=1}^{m} H_i$) from sensors to their respective closest RSUs. Since the two measures are of different unit, they are normalized and a parameter λ from [0, 1] is used to weight the two terms. The denominator of the first term is the sum of the distance by which each of the m sensors can transmit to a sensor (i.e., C_S) and then to an RSU (i.e., C_R). Since each transmission takes at most two hops, m sensors take at most $2m$ hops, which is the denominator of the second term. Constraint (6) computes the distance D_{ij} between sensor i and RSU j. Constraint (7) enforces each RSU to be located at the two sides and the middle island of the road. Since Constraint (9) – (11) enforce that only one of parameters a_{ij}'s and b_{ikj}'s to be one, Constraint (8) computes the distance L_i, which could be the distance from sensor i to RSU j directly or indirectly via sensor k relaying. Constraints (12) enforces that sensor i must fall within the transmission range of RSU j; Constraints (13) enforces that sensor k must fall within the transmission range of RSU j; Constraint (14) enforces that sensor i must fall within the transmission range of a sensor k, and sensor k must fall within the transmission range of RSU j. Constraint (15) computes the total number of hops from each sensor i to its closest RSU. Constraint (16) is the capacity constraint, in which each RSU j can serve at most Q sensors. Constraint (17) computes the distance E_{uvi} between grid point (u, v) and sensor i. Constraint (18) computes the distance F_{uvj} between grid point (u, v) and RSU j. Constraints (19) and (21) enforces that grid point (u, v) must fall within the coverage of sensor i; Constraints (20) and (21) enforces

that grid point (u, v) must fall within the coverage of RSU j. Constraint (22) enforces each grid point of the road to be covered by a sensor or an RSU.

3.2. The proposed approach

This problem is NP-complete because it includes the 2D critical grid coverage problem, which is NP-complete [15]. Hence, this subsection proposes a harmony search algorithm (HSA) [21] for this problem. The HSA is a metaheuristic algorithm that imitates impromptus of multiple musicians, and has been showed to perform better than the genetic algorithm (GA) by simulation on some popular benchmark data sets. For example, the work in [22] solved the set covering problem with the HSA, which performs better than the GA.

The first step of designing this algorithm is to find a way to represent each candidate solution (determined by n parameters) as a harmony, i.e., a vector of n notes in which each note is a parameter of the candidate solution. Consider n musician, each of who decides one of the n notes. At each iteration of the algorithm, each musician plays a note via some special operations inspired from playing music impromptus, and hence, the notes played by all musicians form a new harmony X^{new}. Since each musician may play a note from his/her previous experience, a number of historic harmonies are stored in a so-called harmony memory (*HM*) matrix, which is kept throughout the whole algorithm. If the new harmony X^{new} performs better than the worst harmony in the *HM* matrix, it replaces the worst one. The algorithm repeats the above procedure until the algorithm stops. Finally, the solution corresponding to the best harmony in the *HM* matrix is the final solution.

In what follows, the main components of the proposed HSA for the problem concerned in this subsection are given in detail.

1) Harmony memory matrix

The first step of designing an HSA is to represent a candidate solution as a harmony. A solution to the deployment problem is determined by grid locations of n RSUs on the road. Hence, a harmony is expressed as follows: $(x_1, x_2, \ldots, x_{2n}) = (X_1, Y_1, X_2, Y_2, \ldots X_j, Y_j, \ldots, X_n, Y_n)$, where (X_j, Y_j) is the grid location of RSU j for each $j \in \{1, 2, \ldots, n\}$.

The HSA works on a harmony memory (*HM*) matrix, consisting of *hms* harmonies and their respective costs as follows:

$$HM = \begin{bmatrix} X_1^1 & Y_1^1 & X_2^1 & Y_2^1 & \cdots & X_n^1 & Y_n^1 & c(X^1) \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ X_1^{hms} & Y_1^{hms} & X_2^{hms} & Y_2^{hms} & \cdots & X_n^{hms} & Y_n^{hms} & c(X^{hms}) \end{bmatrix} \quad (23)$$

where $X^k = (X_1^k, Y_1^k, X_2^k, Y_2^k, ..., X_n^k, Y_n^k)$ is the k-th harmony in the matrix, and $c(X^k)$ is its cost for each $k \in \{1, 2, \ldots, hms\}$.

2) Cost evaluation

The concerned problem is a minimization problem. Hence, performance of a harmony (or candidate solution) is evaluated by a cost value based on content of the harmony. The algorithm of evaluating cost of a harmony X^k is given in Algorithm 1.

Algorithm 1 EVALUATE COST(HARMONY X^k)

```
1: Let all penalty counters count_hop, count_cover, count_capacity to be zero
2: for i = 1, 2, …, m do
3:    Find the RSU j that is the closest to sensor i, according to the locations of RSUs determined by
      harmony X^k
4:    Compute the distance L_i from sensor i to RSU j
5:    Compute number of hops H_i from sensor i to RSU j
6:    if H_i > 2 then
7:       L_i = H_i = 0
8:       count_hop = count_hop + 1
9:    end if
10: end for
11: for each grid point of the road do
12:    if the grid point is not covered by any sensor or RSU do
13:       count_cover = count_cover + 1
14:    end if
15: end for
16: for each RSU j do
17:    if number of sensors in the range of RSU j > Q then
18:       count_capacity = count_capacity + 1
19:    end if
20: end for
21: Output the cost value in (24)
```

Algorithm 1 is explained as follows. Since harmony X^k could be infeasible, Line 1 initializes all penalty counters (used for recording the total number of violating each constraint) to be zero. The algorithm consists of three main loops. The first loop (Lines 2 – 10) aims to compute the distance L_i and number of hops H_i from each sensor i to its closest RSU. Since this paper assumes that number of hops cannot exceed two, Lines 6 – 9 check this condition and increase the penalty counter $count_{hop}$ by one. Then, the second loop (Lines 11 – 15) checks whether all grid points on the road are covered, and increases the penalty counter $count_{cover}$ with number of uncovered grid points. Then, the last loop (Lines

16 – 20) checks the QoS capacity constraint, and increases the penalty counter $count_{capacity}$ if the constraint is violated. Finally, Line 21 calculates the cost value $c(X^k)$ of harmony X^k as follows:

$$\frac{\lambda\sum_{i=1}^{m} L_i / (C_R + C_S) + (1-\lambda)\sum_{i=1}^{m} H_i / 2}{m - count_{hop}} + count_{hop} \cdot penalty_{hop}$$

$$+ count_{cover} \cdot penalty_{cover} + count_{capacity} \cdot penalty_{capacity} \quad (24)$$

where $penalty_{hop}$, $penalty_{cover}$, and $penalty_{capacity}$ are penalties for constraint of maximal number of hops, uncovered grid points, and the capacity constraint, respectively, and are set as large values. Since some sensors violate constraint of maximal number of hops, their L_i's and H_i's cannot be computed. Hence, they are not counted in normalizing the objective function, i.e., the first term of the above equation.

The time complexity of Algorithm 1 is as follows:

Lemma 1. Algorithm 1 can be executed in $O(H \cdot W + m + n)$ time.

Proof. The time complexity of Algorithm 1 is analyzed as follows. Line 1 is done in $O(1)$. The loop in Lines 2 – 10 can be done in $O(H + m)$ as follows. Line 3 for each sensor $i \in \{1, 2, ..., m\}$ (i.e., for each iteration) can be done in $O(H)$, because the ordering of positions of all sensors and RSUs is determined by counting three H-length rows of the road in $O(H)$ so that the RSU closest to each sensor i can be found in $O(1)$. Line 5 for each sensor can be done in $O(H)$. The other lines in this loop can be done in $O(m)$. The loop in Lines 11 – 15 can be done in $O(H \cdot W)$ as it counts all grid points of the road surface. The loop in Lines 16 – 20 can be done in $O(n)$ as it counts all RUSs and the information of the number of sensors in the range of each RSU has been recorded in the above loop. Hence, all the three loops in Algorithm 1 can be done in $O(H + m + H \cdot W + n) = O(H \cdot W + m + n)$.

□

3) Generating a new harmony

In each iteration of the HSA main loop, each note x_i^{new} of a new harmony $(x_1^{new}, x_2^{new}, ..., x_{2n}^{new})$ is generated from a combination of the following three possible operations:

- If a random number from [0, 1] is less than the harmony memory considering rate (*HMCR*), a new note x_i^{new} is selected from $\{ x_i^1, x_i^2, \ldots, x_i^{hms} \}$, i.e., the i-th column of the *HM* matrix.

- If the above operation for *HMCR* is executed, the algorithm further checks if a random number from [0, 1] is less than the pitch adjusting rate (*PAR*). If true, there are two cases as follows. If i is odd (i.e., x_i^{new} is a row location of the road), then x_i^{new} is adjusted slightly by increasing or decreasing $H/2$

but it cannot exceed the lower bound 0 and the upper bound H, i.e., $x_i^{new} = \max\{x_i - H/2, 0\}$ or $\min\{x_i + H/2, H\}$. Similarly, if i is even (i.e., x_i^{new} is a column location of the road), then $x_i^{new} = \max\{x_i - bw, 0\}$ or $\min\{x_i + bw, \mathrm{W}\}$, where bw denotes bandwidth, which is a given parameter of the algorithm.

- If the operation for *HMCR* is not executed, a new note x_i^{new} is generated randomly within its possible range. If i is odd (i.e., x_i^{new} is a row location of the road), then x_i^{new} is a random number from $\{0, H/2, H\}$; otherwise (i.e., x_i^{new} is a column location of the road), x_i^{new} is a random number from $\{0, 1, \ldots, W\}$.

4) Updating the harmony memory matrix

After a new harmony $X^{new} = (x_1^{new}, x_2^{new}, ..., x_n^{new})$ is generated, the algorithm finds the worst harmony X^{worst} in the *HM* matrix and compares which of them performs better in terms of their costs. If X^{new} performs better, X^{new} replaces X^{worst} in the *HM* matrix; otherwise, it is discarded.

5) Algorithm

The proposed HSA for the HVSN deployment problem concerned in this subsection is given in Algorithm 2, which is explained as follows. Lines 1 and 2 generate the *HM* matrix and evaluate cost of each harmony in the matrix. Then, the main loop (Lines 3 – 26) iterates with an iteration number k until it exceeds the maximal iteration number η. In the loop, Line 5 considers each of the $2n$ notes of the new harmony $X^{new} = (x_1^{new}, x_2^{new}, ..., x_{2n}^{new})$. Line 6 checks the condition of *HMCR*. If true, Line 7 executes the operation for *HMCR*. Line 8 checks the condition of *PAR*. If true, Lines 9 – 13 execute the operation for *PAR*. If the condition of *HMCR* does not hold, Lines 16 – 20 execute the operation for randomly generating a note. Then, the cost of the generated new harmony is evaluated in Line 23 and the new harmony is accepted or rejected in Line 24. Finally, Line 27 outputs the solution corresponding to the best harmony in the *HM* matrix as the final solution.

The time complexity of Algorithm 2 is as follows:

Theorem 1. Algorithm 2 can be executed in $O(\eta \cdot (H \cdot W + m + n + hms))$ time, when $hms < \eta$.

Proof. The time complexity of Algorithm 2 is analyzed as follows. The *HM* matrix with size $2n \cdot hms$ in Line 1 can be initialized randomly in $O(2n \cdot hms)$. By Lemma 1, costs of all hms harmonies can be computed in $O(hms \cdot (H \cdot W + m + n))$.

The loop in Lines 5 – 22 can be done in $O(2n)$ because each line of each iteration of the loop with $2n$ iterations can be done in $O(1)$. Line 23 is done in $O(H \cdot W + m + n)$ by Lemma 1. Line 24 can be done in $O(hms)$ by considering costs of all *hms* harmonies in the *HM* matrix. Hence, the loop with η iterations in Lines 3 – 26 can be done in $O(\eta \cdot (2n + H \cdot W + m + n + hms)) = O(\eta \cdot (H \cdot W + m + n + hms))$.

In general, number of harmonies *hms* is smaller than number of iterations, the total time of initialization and the main loop can be done in $O(\eta \cdot (H \cdot W + m + n + hms))$, as required.

□

Since m sensors and n RSUs are deployed along three H-length rows, thus $m + n < 3H$. Hence, Algorithm 2 can further be executed in $O(\eta \cdot (H \cdot W + hms))$ time.

Algorithm 2 HSA FOR DEPLOYING HVSN ON A GRID ROAD

1: Randomly generate the *hms* harmonies in the *HM* matrix
2: Evaluate costs of the *hms* harmonies
3: Initialize the iteration number $k = 1$
4: **while** $k \leq$ the maximal iteration number η **do**
5: **for** $i = 1, 2, \ldots, 2n$ **do**
6: **if** *rand*(0, 1) < *HMCR* **then**
7: x_i^{new} = randomly selected from $\{x_i^1, x_i^2, \ldots, x_i^{hms}\}$
8: **if** *rand*(0, 1) < *PAR* **then**
9: **if** i is odd **then**
10: $x_i^{new} = \max\{x_i - H/2, 0\}$ or $\min\{x_i + H/2, H\}$
11: **else**
12: $x_i^{new} = \max\{x_i - bw, 0\}$ or $\min\{x_i + bw, W\}$
13: **end if**
14: **end if**
15: **else**
16: **if** i is odd **then**
17: x_i^{new} = a random number from $\{0, H/2, H\}$
18: **else**
19: x_i^{new} = a random number from $\{0, 1, \ldots, W\}$
20: **end if**
21: **end if**
22: **end for**
23: Evaluate cost of the new harmony $(x_1^{new}, x_2^{new}, ..., x_{2n}^{new})$
24: Accept or reject the new harmony according to its cost
25: Increase the iteration number k
26: **end while**
27: Output the solution corresponding to the harmony with the minimal cost in the *HM* matrix

4. The Deployment Problem for a Dense HVSN

This section first describes the problem of deploying a dense HVSN, and then provides an improved HSA approach for this problem.

4.1. Problem description

Consider an HVSN in which a large number of sensors have been located densely and nonuniformly in a rectangular deployment area, because generally sensors are installed densely on the road segments with heavy traffic flow, e.g., interchange exits and accessory roads. Hence, it is of interest to deploy RSUs to serve sensors in a dense HVSN. Different from the deployment problem in the last section that allows RSUs to be deployed only along the two sides and the middle island of the road, the problem of deploying a dense HVSN allows RSUs to be deployed at any location in the rectangular deployment area.

Consider a two-dimensional rectangular geographical area of size $H \times W$, as illustrated in Fig. 3, in which sensors are located densely and nonuniformally. The problem concerned in this section is to determine locations of n RSUs to serve those sensors in the area, so that the total distance and the total number of hops from sensors to their respective closest RSU are minimized, when the QoS constraints for the maximal number of hops and capacity are satisfied. Note that each sensor communicates with an RSU directly (i.e., a sensor is covered by an RSU in Fig. 3) or indirectly via another sensor relaying (i.e., a sensor is covered by another sensor that is cover by an RSU in Fig. 3); while each RSU can be connected to the Internet.

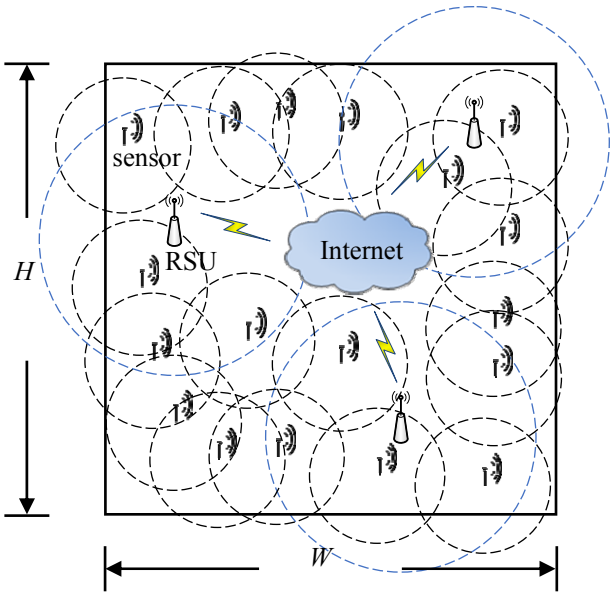

Fig. 3. Illustration of a dense HVSN.

The mathematical programming model of the concerned problem is the same as that in the last section (see Objective (5) and Constraints (6) – (22)) except for the following decision variable setting:

- (X_j, Y_j): Location of RSU j, which could be any location of the rectangular $H \times W$ area, and is the decision variable of this problem.

Hence, Constraint (7) in the model of the last section is replaced as follows:

$$X_j \in [0, H], Y_j \in [0, \mathrm{W}], \forall j \in \{1, 2, \ldots, n\} \tag{25}$$

That is, the model in the last section is an integer programming model; while the model in this section is a mixed-integer programming model.

4.2. The proposed approach

This subsection proposes a geometric selective harmony search (GSHS) [23] for the problem. The GSHS is an improved version of HSA, by making use of some evolutionary operators from the GA. It includes the following two main improvements. The first improvement works on the *HMCR* operation. Different from the legacy HSA that randomly selects a note from the *HM* matrix to generate a new harmony, the GSHS imitates the GA to use the tournament selection to choose harmonies for generating a better new harmony and balancing the convergence time and the solution-searching diversity. The second improvement works on the *PAR* operation. The GSHS introduces a mutation equation with geometric parameters for adjusting harmonies, different from the HSA that adopts parameter *bw*. By doing so, the local search ability of the new harmony is increased, and premature convergence can be avoided.

Note that the decision variables in this section are continuous, different from those in the last section. Although the HSA can also be applied to continuous decision variables, the GSHS performs better than the HSA [23]. Additionally, the GSHS includes crossover and mutation schemes from the GA, so as increase the solution diversity to avoid local optimal solutions. Hence, this section applies the GSHS to solve the concerned problem. The main components of the GSHS are detailed as follows.

1) Harmony memory matrix

A solution to the deployment problem for a dense HVSN is determined by (x, y)-coordinates of n RSUs within a $H \times W$ area, i.e., a harmony is expressed as follows: $(x_1, x_2, \ldots, x_{2n}) = (X_1, Y_1, X_2, Y_2, \ldots X_j, Y_j, \ldots, X_n, Y_n)$, where (X_j, Y_j) is the (x, y)-coordinate of RSU j for each $j \in \{1, 2, \ldots, n\}$. The *HM* matrix is the same as Equation (23), but the main difference is that all the entries in the matrix here are real numbers, not integers.

2) Cost evaluation

The cost of a harmony is expressed as follows:

$$\frac{\lambda\sum_{i=1}^{m} L_i / (C_R + C_S) + (1-\lambda)\sum_{i=1}^{m} H_i / 2}{m - count_{hop}} + count_{hop} \cdot penalty_{ho} + count_{capacity} \cdot penalty_{capacit} \quad (26)$$

All terms in the above formula can be obtained by Algorithm 1. Note that the difference of Equation (26) from Equation (24) is that Equation (26) is lacking of the penalty for uncovered grid points, because the dense HVSN is distributed on a wide area and restricted to roads so that it is impossible to cover the whole area.

3) Generating a new harmony

The three operations used to generate a new harmony are as follows:

- The *HMCR* operation: Let *ts* denote the tournament size, which is a parameter used in the tournament selection. In this paper, *ts* is set to 2. The tournament selection selects the harmony with the best cost from two (or *ts*) harmonies selected randomly from the *HM* matrix. Hence, after two tournament selections, two new harmonies x_i^{new1} and x_i^{new2} are selected. Then, a new harmony x_i^{new} is generated by $x_i^{new} = \alpha \cdot x_i^{new1} + (1-\alpha) \cdot x_i^{new2}$, which is a linear combination of x_i^{new1} and x_i^{new2}, where α is a number from [0, 1]. This paper sets $\alpha = 0.5$.

- The *PAR* operation: Let *ms* denote the mutation step, which is a parameter in this operation. This operation adjusts $x_i^{new} = x_i^{new} \pm ms \cdot rand(0,1) \cdot x_i^{new}$, i.e., it is increased or decreased with a random small amount of the mutation step.

- The random generation operation: Let min_i and max_i denote the minimal and maximal notes in the *i*-th column of the *HM* matrix, respectively, in $\{ x_i^1, x_i^2, \ldots, x_i^{hms} \}$. Then, a new note x_i^{new} is generated as follows: $x_i^{new} = min_i + rand(0, 1) \cdot (max_i - min_i)$, i.e., a random value between the minimal and maximal notes in the *i*-th column of the *HM* matrix.

4) Algorithm

The proposed GSHS algorithm for the deployment problem concerned in this section is given in Algorithm 3, which is almost the same with Algorithm 2 except for the three operations (Lines 7 – 9, 12, 14, and 18) for generating a new harmony and the cost evaluation. Based on a similar analysis to Algorithm 2, Algorithm 3 can also be executed in $O(\eta \cdot (H \cdot W + m + n + hms))$ time.

Algorithm 3 GSHS FOR DEPLOYING DENSE HVSN

Randomly generate the *hms* harmonies in the *HM* matrix
Evaluate costs of the *hms* harmonies
Initialize the iteration number $k = 1$
while $k \le$ the maximal iteration number η **do**
  for $i = 1, 2, \ldots, 2n$ **do**
    if $rand(0, 1) < HMCR$ **then**
      Select a note x_i^{new1} from the *HM* matrix via tournament selection
      Select a note x_i^{new2} from the *HM* matrix via tournament selection
      $x_i^{new} = \alpha \cdot x_i^{new1} + (1-\alpha) \cdot x_i^{new2}$
      if $rand(0, 1) < PAR$ **do**
        if *i* is odd **then**
          $x_i^{new} = \max\{x_i^{new} - ms \cdot rand(0,1) \cdot x_i^{new}, 0\}$ or $\min\{x_i^{new} + ms \cdot rand(0,1) \cdot x_i^{new}, W\}$
        else
          $x_i^{new} = \max\{x_i^{new} - ms \cdot rand(0,1) \cdot x_i^{new}, 0\}$ or $\min\{x_i^{new} + ms \cdot rand(0,1) \cdot x_i^{new}, H\}$
        end if
      end if
    else
      $x_i^{new} = min_i + rand(0, 1) \cdot (max_i - min_i)$
    end if
  end for
  Evaluate cost of the new harmony $(x_1^{new}, x_2^{new}, ..., x_{2n}^{new})$
  Accept or reject the new harmony according to its cost
  Increase the iteration number *k*
end while
Output the solution corresponding to the harmony with the lowest cost in the *HM* matrix

5. Implementation and Experimental Results

This section implements the HSA for the deployment problem for a grid road in Section 3 and the GSHS for the deployment problem for a dense HVSN in Section 4, and conducts a comprehensive experimental analysis.

5.1. Experimental setting

The experiments are conducted on a PC with an Intel i7-3770 CPU 3.40 GHz 3.40 GHz and 16 GB RAM. The HSA and GSHS are implemented in C++ programming language. The parameter setting used in deploying an HSVN on a grid road is given in Table 2; that used in deploying a dense HVSN is given in Table 3. Additionally, nonuniform distribution of sensor positions in the experiments applies the centralized distribution in spatial distribution [24], [25], which aims to make samples distributed at the center of this area to some degree. More specifically, (*x*, *y*)-coordinate of each sensor satisfies the following probability density function:

$$\begin{cases} f_X(x) = (\dfrac{2\beta - 2}{W}) \cdot \left| x - \dfrac{W}{2} \right| + 1, & \text{for } 0 \le x \le W; \\ f_Y(y) = (\dfrac{2\beta - 2}{H}) \cdot \left| y - \dfrac{H}{2} \right| + 1, & \text{for } 0 \le y \le H. \end{cases} \tag{27}$$

where β is the centralization coefficient in range [0, 1]. When β is larger, the distribution is more uniform; and it is a uniform distribution when $\beta = 1$. Conversely, when $\beta = 0$, almost no samples are distributed at the range borders. Our experiments sets $\beta = 0.5$.

Table 2. The parameter setting used in deploying an HSVN on a grid road.

Parameter	Value
H	10,12,15
W	300,400,500
Number of RSUs	10,16,25
Number of sensors	50,80,125
Coverage of an RSU (C_R)	80
Coverage of a sensor (C_S)	15
$penalty_{hop}$	100
$penalty_{cover}$	100
$penalty_{capacity}$	100

Table 3. The parameter setting used in a dense HVSN.

Parameter	Value
Height H	300,400,500
Width W	300,400,500
Number of RSUs	10,25,35
Number of sensors	50,75,125
Coverage of an RSU (C_R)	80
Coverage of a sensor (C_S)	15
$penalty_{hop}$	100
$penalty_{capacity}$	100

Table 2 lists the parameter setting used in deploying an HSVN on a grid road. Three instances with different road sizes are set as follows: small ($H = 10$, $W = 300$), middle ($H = 12$, $W = 400$), and large ($H = 15$, $W = 500$). Table 3 lists the parameter setting used in a dense HVSN. Three instances with different road sizes are set as follows: small ($H = 300$, $W = 300$), middle ($H = 400$, $W = 400$), and large ($H = 500$, $W = 500$). In the two different-dimensional road settings, coverage of an RSU is set as 80, coverage of a sensor is 15, and all penalties are set as 100.

5.2. Sensitivity analysis of parameters used in the HSA

The work in [22] proposed that using a larger *HMCR* value leads to a harmony with more diversity, which could lead to a better solution. Hence, we conduct experimental analysis in Fig. 4(a), which uses a fixed *PAR* value and different *HMCR* values (i.e., 0.5, 0.6, 0.7, 0.8, and 0.9). From Fig. 4(a), the result

using *HMCR* = 0.9 performs the best. Hence, *HMCR* = 0.9 in later HSA experiments. On *PAR*, the work in [22] proposed that using a smaller *PAR* value could avoid premature convergence. Hence, we conduct experimental analysis in Fig. 4(b), which uses a fixed *HMCR* value and different *PAR* = 0.1, 0.2, 0.3, 0.4, and 0.5. From Fig. 4(b), the result with *PAR* = 0.2 performs the best. Hence, *PAR* = 0.2 in later HSA experiments. Using similar sensitivity analysis, the most appropriate parameter setting used in the HSA is found and given in Table 4.

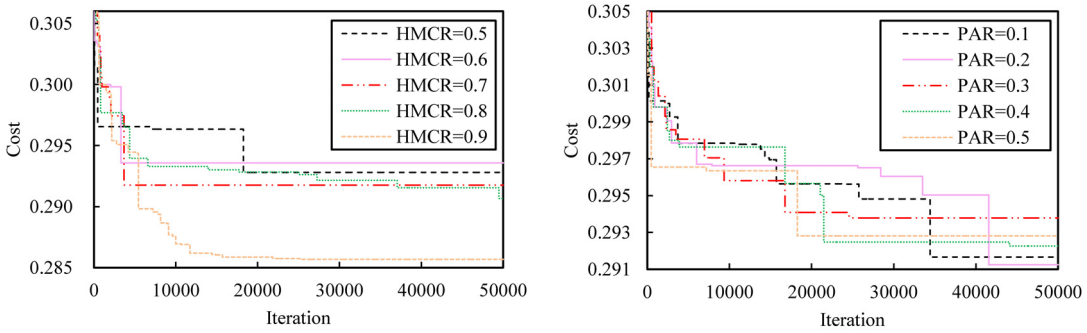

Fig. 4. Result of executing HSA on instances with (a) different *HMCR* values and (b) different *PAR* values.

Table 4. The parameter setting used in the HSA.

Parameter	Value
Harmony memory size (*hms*)	100
Harmony memory considering rate (*HMCR*)	0.9
Pitch adjusting probability (*PAR*)	0.2
Bandwidth (*bw*)	1
Number of iterations (η)	50000

5.3. Sensitivity analysis of parameters used in the GSHS

Using similar sensitivity analysis detailed in the above subsection, we conduct experimental analysis in Fig. 5(a), which uses a fixed *PAR* value and different *HMCR* values (i.e., 0.5, 0.6, 0.7, 0.8, and 0.9) and in Fig. 5(b), which uses a fixed HMCR value and different PAR values (i.e., 0.1, 0.2, 0.3, 0.4, and 0.5). *HMCR* = 0.9 and *PAR* = 0.4 are set in later experiments for GSHS. Note that the GSHS has one more parameter *ms*. The work in [23] proposed that a larger *ms* value could avoid local optimal solutions, increase diversity, and avoid premature convergence. Hence, using similar sensitivity analysis, we let *ms* = 0.07 for later experiments for GSHS (Fig. 5(c)). Finally, the most appropriate parameter setting used in the GSHS is found and given in Table 5.

5.4. Experimental analysis for deploying an HVSN grid road

For the HVSN deployment problem for a road represented as grid points, Fig. 5(d) gives the results of executing HSA on instances with numbers of RSUs (in which $H = 10$ and $W = 300$). From Fig. 5(d), the costs for the cases with 10, 15, and 20 RSUs look similar; the case with 5 RSUs is feasible, but has a larger cost. Since cost of installing more RSUs is expensive, and 15 and 20 RSUs do not improve too much from Fig. 5(d), we apply 10 RSUs in the remaining experiments.

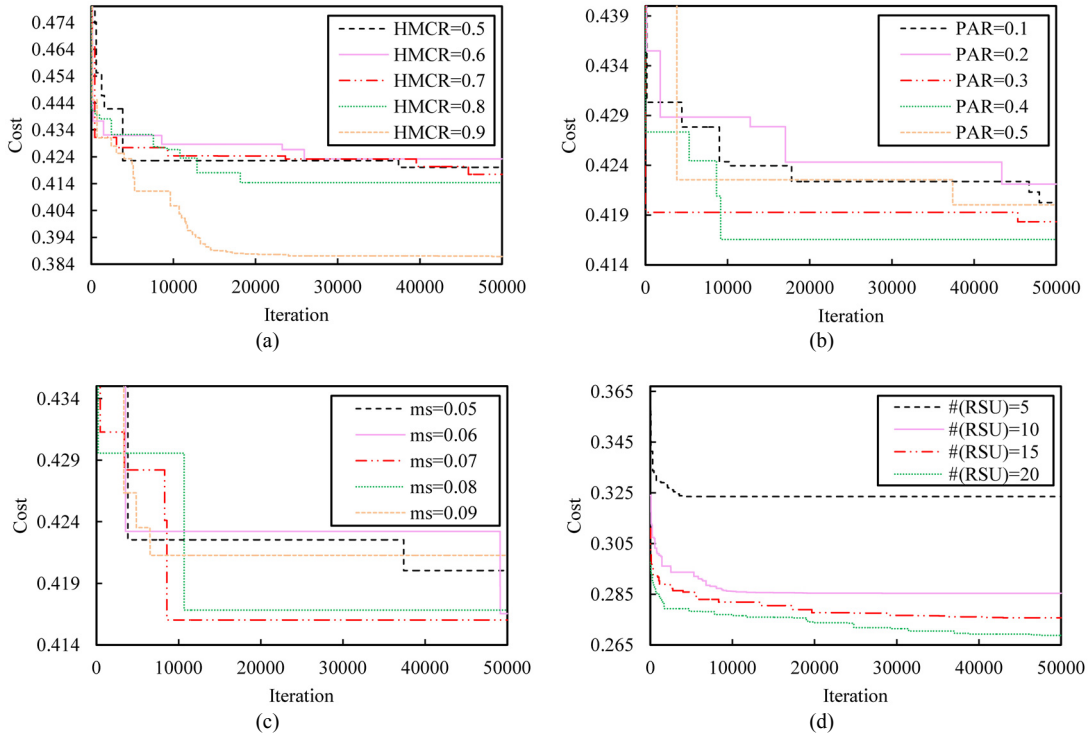

Fig. 5. Result of executing GSHS on instances with (a) different *HMCR* values, (b) different *PAR* values, (c) different *ms* values, and (d) different numbers of RSUs.

Table 5. The parameter settings used in the GSHS.

Parameter	Value
Harmony memory size (*hms*)	100
Harmony memory considering rate (*HMCR*)	0.9
Pitch adjusting probability (*PAR*)	0.4
Number of iterations (η)	50000
Tournament size (*ts*)	2
Mutation step (*ms*)	0.07

Next, we analyze the change of costs in three instances with different road sizes: small ($H = 10$, $W = 300$), middle ($H = 12$, $W = 400$), and large ($H = 15$, $W = 500$). The box-and-whisker plots of running 100 times of HSA on the three instances are given in Fig. 6, and their statistics are given in Table 6. From Fig. 6, we observe that the larger the problem size is, the larger the cost (i.e., Mean, Min, Max). Additionally, all instances have no penalty cost, i.e., all the solutions are feasible. From Table 6, we observe that the

larger the problem size is, the larger the CPU time is.

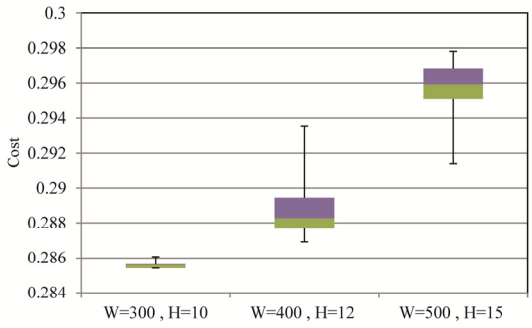

Fig. 6. Box-and-whisker plots of running 100 times of HSA on three different-size instances.

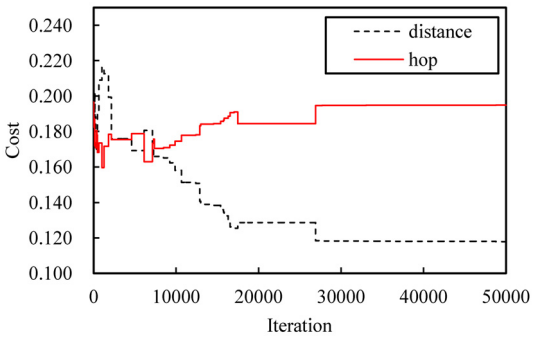

Fig. 7. Plots of costs of the two terms (i.e., distance and number of hops) versus number of iterations for an HSA result.

Table 6. Statistics for running 100 times of HSA on three instances.

	Mean	Min	Max	StdDev	CPU time (s)
$H = 10$, $W = 300$	0.28557	0.28541	0.28607	1.3486×10^{-4}	4.041
$H = 12$, $W = 400$	0.28882	0.28690	0.29355	1.6825×10^{-3}	10.252
$H = 15$, $W = 500$	0.29588	0.29135	0.29782	1.1052×10^{-3}	24.597

Table 7 gives the result of Wilcoxon signed rank test on the 100 HSA results for three instances. Note that the Wilcoxon signed rank test is nonparametric (i.e., the assumption of a normal distribution of the same variance is not required), and it aims to test difference of paired samples. From Table 7, the p-value of each pair of instances is less than 0.05, meaning that the differences between results of the three instances are remarkable statistically.

Table 7. Wilcoxon signed rank test of the HSA results for three instances.

P-value	$H = 10$, $W = 300$	$H = 12$, $W = 400$	$H = 15$, $W = 500$
$H = 10$, $W = 300$	0	2.0261×10^{-34}	3.6724×10^{-34}
$H = 12$, $W = 400$	–	0	2.0263×10^{-34}
$H = 15$, $W = 500$	–	–	0

Costs of the total distance and the total number of hops at different iterations for the small-size instance are plotted in Fig. 7. From Fig. 7, except for the initial stage, as number of iterations decreases, the cost for the total distance decreases; and the cost for the total number of hops increases. That is, the two dynamics go inversely. We speculate that the road grid height (i.e., the H parameter) is so small that a very small cost for the total distance can be found finally in Fig. 7. Another observation from Fig. 7 is that the decreasing range of the cost for the total distance is greater than that for the total number of hops.

We speculate that the cost for number of hops is restricted by the QoS constraint so that it could not decrease soon.

5.5. Experimental analysis for deploying a dense HVSN

For the problem of deploying a dense HVSN, Fig. 8 gives the results of executing GSHS on the instances with different numbers of RSUs (in which $H = W = 300$). We apply 10 RSUs in the remaining experiments, based on the same reason in the last subsection. Next, the box-and-whisker plots of running 100 times of GSHS on three instances with roads of small ($H = W = 300$), middle ($H = W = 400$), and large ($H = W = 500$) sizes are given in Fig. 9 and their statistics are given in Table 9. We have the same conclusion with that in the last subsection. From Fig. 9, we observe that the larger the problem size is, the larger the cost (i.e., Mean, Min, Max) is. Additionally, all instances have no penalty cost, i.e., all the solutions are feasible. From Table 8, we observe that the larger the problem size is, the larger the CPU time is. Table 9 gives the Wilcoxon signed rank test on 100 GSHS results for three instances, in which each p-value value is less than 0.05, meaning that the differences between results of the three instances are remarkable statistically.

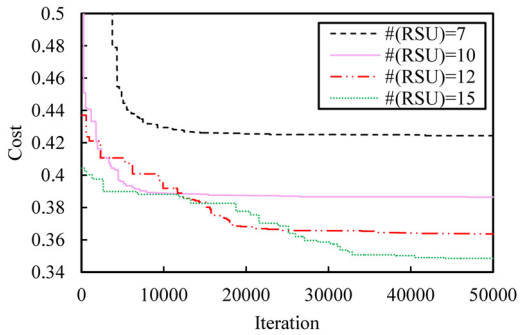

Fig. 8. Results of executing GSHS on instances with different numbers of RSUs.

Costs of the total distance and the total number of hops at different iterations for the small-size instance are plotted in Fig. 10. The plots in Fig. 10 are quite different from those in Fig. 7. From Fig. 10, cost for the total distance is much greater than cost for number of hops. The reason could be that size of the deployment area (i.e., $H \times W$) is so large (as compared to Fig. 7) that cost for the total distance is large. As for cost for number of hops, the reason why it is smaller is that it is restricted to the QoS constraint.

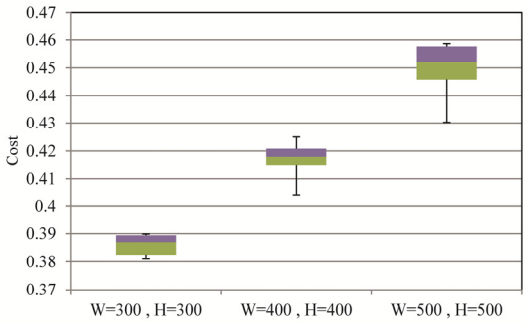

Fig. 9. Box-and-whisker plots of running 100 times of GSHS on three different-size instances.

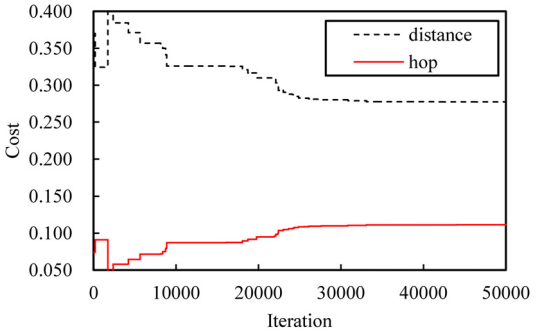

Fig. 10. Plots of costs of the two terms (i.e., distance and number of hops) versus number of iterations for a GSHS result.

Table 8. Statistics for running 100 times of GSHS on three instances.

	Mean	Min	Max	StdDev	CPU time (s)
$H = W = 300$	0.38641	0.38095	0.39221	3.2998×10^{-3}	10.025
$H = W = 400$	0.41775	0.40397	0.43049	4.9241×10^{-3}	20.141
$H = W = 500$	0.45202	0.43019	0.47518	9.4160×10^{-3}	39.696

Table 9. Wilcoxon signed rank test of the GSHS results for three instances.

P-value	$H = W = 300$	$H = W = 400$	$H = W = 500$
$H = W = 300$	0	2.5621×10^{-34}	2.5620×10^{-34}
$H = W = 400$	–	0	1.3201×10^{-34}
$H = W = 500$	–	–	0

6. Conclusion

This paper has investigated different-dimensional deployment problems in HVSNs, with objective to minimize the total distance and the total number of hops from sensors to the closest RSU, under constraints of the maximal number of hops and capacity. For deploying HVSNs on one-dimensional roads, a linear programing model is provided as the solution. Next, the two-dimensional deployment problems are classified into two categories: the deployment problem for covering all grid points of a road; and the problem of deploying a dense HVSN. This paper first establishes mathematical programming models for the two problems, and then proposes an HSA for the former problem and a GSHS for the latter problem. The two algorithms are implemented and their performance is analyzed in detail. Statistical analysis show significant differences on executing the proposed approaches for different-size instances of different problem settings concerned. A potential future work is to integrate design of lower layers in our problem models. Additionally, roads may be three-dimensional when overhead loads are concerned, and hence, it is of interest to deploy three-dimensional HVSNs.

Acknowledgements

The authors thank the anonymous referees for comments that improved the content as well as the presentation of this paper. This work has been supported in part by MOST 104-2221-E-009-134-MY2, Taiwan.

References

[1] Li P, Huang X, Fang Y, Lin P (2007) Optimal placement of gateways in vehicular networks. *IEEE Transactions on Vehicular Technology* 56(6), pp. 3421-3430.

[2] Wu TJ., Liao W, Chang CJ (2012) A cost-effective strategy for road-side unit placement in vehicular networks. *IEEE Transactions on Communications* 60(8), pp. 2295-2303.

[3] Lochert C, Scheuermann B, Wewetzer C, Luebke A, Mauve M (2008) Data aggregation and roadside unit placement for a VANET traffic information system. In: *Proceedings of 5th ACM International Workshop on Vehicular Inter-NETworking*, pp. 58-65, ACM Press.

[4] Cavalcante ES, Aquino ALL, Pappa GL, Loureiro AAF (2012) Roadside unit deployment for information dissemination in a VANET: An evolutionary approach. In: *Proceedings of GECCO'12*, pp. 27-34, ACM Press.

[5] Liya X, Chuanhe H, Peng L, Junyu Z (2013) A randomized algorithm for roadside units placement in vehicular ad hoc network. In: *Proceedings of MSN* 2013, pp. 193-197, IEEE Press.

[6] Aslam B, Zou CC (2011) Optimal roadside units placement along highways. In: *Proceedings of 2011 IEEE Consumer Communications and Networking Conference*, pp. 814-815, IEEE Press.

[7] Ng SC, Zhang W, Zhang Y, Yang Y, Mao G (2011) Analysis of access and connectivity probabilities in vehicular relay networks. *IEEE Journal on Selected Areas in Communications* 29(1), pp. 140-150.

[8] Sou S, Tonguz OK (2011) Enhancing VANET connectivity through roadside units on highways. *IEEE Transactions on Vehicular Technology* 60(8), pp. 3586-3602.

[9] Patil P, Gokhale A (2012) Maximizing vehicular network connectivity through an effective placement of road side units using Voronoi diagrams. In: *Proceedings of MDM* 2012, pp. 274-275, IEEE Press.

[10] Hussain Rehman OM, Bourdoucen H, Ould-Khaoua M (2015) Forward link quality estimation in VANETs for sender-oriented alert messages broadcast. *Journal of Network and Computer Applications* 58, pp. 23-41.

[11] Weingatner E, Kargl F (2007) *A Prototype Study on Hybrid Sensor-Vehicular Networks*. Technical report, RWTH-Aachen, Aachen, Germany.

[12] Qin R, Li Z, Wang Y, Lu X, Zhang WS (2010) An integrated network of roadside sensors and vehicles for driving safety: Concept, design and experiments. In: *Proceedings of 2010 IEEE International Conference on Pervasive Computing and Communications*, pp. 79-87, IEEE Press.

[13] Murthy GR, Piran MJ (2011) *Instantaneous Accident Detection and Notification System*, 3780/CHE/2011 A, November, 2011.

[14] Rebai M, Khoukhi L, Snoussi H, Hnaien F (2012) Optimal placement in hybrid VANETs-sensors networks. In: *Proceedings of IEEE Wireless Advanced*, pp. 54-57, IEEE Press.

[15] Ke WC, Liu BH., Tsai MJ (2011) The critical square grid coverage problem in wireless sensor networks is NP-complete. *Computer Networks* 55, pp. 2209-2220.

[16] Lin C, Deng D (2015) Optimal two-lane placement for hybrid VANET-sensor networks. *IEEE Transactions on Industrial Electronics*, in press.

[17] Bejerano Y (2004) Efficient integration of multihop wireless and wired networks with QoS constraints. *IEEE/ACM Transactions on Networking* 12(6), pp. 215-226.

[18] Moussaoui A, Boukeream A (2015) A survey of routing protocols based on link-stability in mobile ad hoc networks. *Journal of Network and Computer Applications* 47, pp. 1-10.

[19] Zheng J, Wu Y, Xu Z, Lin X (2014) A reliable routing protocol based on QoS for VANET. In: *Proceedings of 2014 IEEE Conference on Advanced Infocomm Technology*, pp. 21-28, IEEE Press.

[20] Aoun B, Boutaba R, Iraqi Y, Kenward G (2006) Gateway placement optimization in wireless mesh networks with QoS constraints. *IEEE Journal on Selected Areas in Communications* 24(11), pp. 2127-2136.

[21] Geem ZW, Kim JH (2001) A new heuristic optimization algorithm: Harmony search. *Simulation* 76, pp. 60-68.

[22] Nezhad SE (2010) Solving k-coverage problem in wireless sensor networks using improved harmony search. In: *Proceedings of IEEE Broadband, Wireless Computing, Communication and Applications*, pp. 49-55, IEEE Press.

[23] Castelli M, Silva S, Manzoni L, Vanneschi L (2014) Geometric selective harmony search. *Information Sciences* 279, pp. 468-482.

[24] Li C, Wang L, Sun T, Yang S, Gan X, Yang F, Wang X (2014) Topology analysis of wireless sensor networks based on nodes' spatial distribution. *IEEE Transactions on Wireless Communications*

13(5), pp. 2454-2453.

[25] Senouci MR, Mellouk A, Senouci H, Aissani A (2012) Performance evaluation of network lifetime spatial-temporal distribution for WSN routing protocols. *Journal of Network and Computer Applications* 35, pp. 1317-1328.