Genetic Programming for Lifetime Maximization in Wireless Sensor Networks with a Mobile Sink

Ying Li[1], Zhixing Huang[1], Jinghui Zhong[1](✉), and Liang Feng[2]

[1] School of Computer Science and Engineering, South China University of Technology, Guangzhou, China
jinghuizhong@scut.edu.cn
[2] College of Computer Science, Chongqing University, Chongqing, China

Abstract. Maximizing the lifetime of Wireless Sensor Network (WSN) with a mobile sink is a challenging and important problem that has attracted increasing research attentions. In the literature, heuristic based approaches have been proposed to solve the problem, such as the Greedy Maximum Residual Energy (GMRE) based method. However, existing heuristic based approaches highly rely on expert knowledge, which makes them inconvenient for practical applications. Taking this cue, in this paper, we propose an automatic method to construct heuristic for sink routing based on Genetic Programming (GP) approach. Empirical study shows that the proposed method can generate promising heuristics that achieve superior performance against existing methods with respect to the global lifetime of WSN.

1 Introduction

Wireless Sensor Network (WSN) is a network formed by sensor nodes (SNs) that communicate via wireless signals [1,2]. Compared with wired network, WSN has the advantage of less cost, higher robustness, plummy traceability and portability [3]. Owning to these advantages, WSN has been used in a range of real-world applications, such as earthquake monitoring [4], communication supervising [5], climate changing [6], smart home [7] and smart city design [8], etc.

In a WSN, the SNs are usually battery-powered and it is often difficult or even impossible to recharge the SNs. Thus, maximizing the lifetime of WSN is an important issue in many WSN applications [9–11]. In the literature, one of the effective and popular techniques to prolong the lifetime of WSN is using a mobile sink to collect information from the network. In particular, by moving the sink to proper locations (i.e., sink sites) of the network, the energy consumption of the network can be reduced and its lifetime can be prolonged [12,13]. However, how to schedule the moving paths of the sink, i.e., the Mobile Sink Scheduling Problem (MSSP), become a challenging and hop research topic recently.

Over the past decades, a number of efforts have been proposed in the literature to solve the MSSP. Existing works mainly adopted centralized approaches

The first author and the second author contributed equally to this work.

Y. Shi et al. (Eds.): SEAL 2017, LNCS 10593, pp. 774–785, 2017.
https://doi.org/10.1007/978-3-319-68759-9_63

which require knowing the global information of the WSN such as network topology, communication costs, etc. For example, Wang et al. [14] proposed a linear programming optimization model to optimize the moving path of the sink. Similar work that uses LP methods to solve the MSSP can be found in [15,16]. Zhong and Zhang [17] proposed an Ant Colony Optimization approach to solve the MSSP. The centralized methods are capable of finding the globally optimal or near global optimal solutions, but they require large memory and long computational time. To overcome these limitations, decentralized approach have also been proposed recently. The key idea is to design heuristic rules to schedule the movements of the sink step-by-step. For example, Basagni et al. [18] proposed a Greedy maximum Residual Energy (GMRE) which moves the sink in rounds. At the end of each round, the sink makes a constant movement towards the node that has the most residual energy. Typically, the heuristic rules are designed in a fairly ad-hoc manner, and relies on expert intuition. However, these intuition often mislead the sink routing since the WSN is a complex system and the relationship between the heuristic rule and the final network lifetime are unintuitive. How to automatically design effective heuristic rule is still challenging.

To address the above issue, in this paper, a GP based approach is proposed to solve the MSSP. Our key idea is using GP to automatically learn a heuristic function that can dynamically schedule the movements of the sink node. GP is an evolutionary algorithm (EA) that solves user-defined tasks by the evolution of computer programs. It has been shown to be highly efficient in many applications, including symbolic regression [19,20], Job Shop Scheduling Problem (JSP) [21] and workflow scheduling [22], etc. However, to the best of our knowledge, there is little work in the literature that uses GP for the sink mobility optimization. Thus, this paper makes an attempt to apply the GP to solve this kind of problem.

In our proposed method, the "sense-think-act" paradigm is adopted to schedule the sink. In particular, the sink node detects the environment features first, and a high-level heuristic (HH) is then utilized to determine the moving strategy based on the input features. Finally, the sink will either remain stationary or move to the next location according to the strategy. The above steps are repeated until the WSN become "dead". To build effective heuristic, four environment features are defined and the Self-Learning Gene Expression Programming (SL-GEP) [23] is utilized to construct a HH. These environment features are combined by the common numerical functions. The proposed method is tested on WSNs with different scales and deployment strategies. The empirical results demonstrated that our method can achieve superior performance over several other approaches, in terms of maximizing the network lifetime.

2 Problem Definition

In this paper, we consider that there are two kinds of nodes in WSN: sink node and sensor node (SN). The SNs are static, battery-powered and deployed over the monitoring region to sense the physical world, while the sink node is mobile and used to collect data from the SNs. The sink node has unlimited power and move over the candidate locations (i.e. sink sites). According to [24], we evenly divide the monitoring region into discrete grids and use the intersection points of the grids as sink sites. Figure 1 shows a typical example of the WSN with 19 SNs and 35 sink sites.

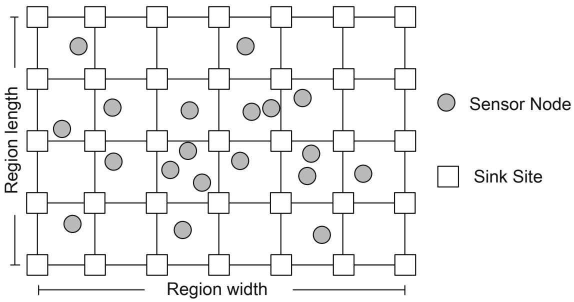

Fig. 1. A WSN with a single sink.

The maximum communication radius of the sink node is R. That is, a SN can communicate with the sink node in one hop if its distance to the sink node is less than or equal to R. If a SN is out of the communication radius of the sink node, its data can send to the sink node via a multi-hop manner. To minimize the energy consumption, a minimum spanning tree (MST) is constructed for message routing, whose root is the sink and nodes are SNs. Every sink site is associated with a corresponding MST.

As in [25], the energy consumption of transmitting one package data is given by:

$$C_{ij} = a * d_{ij}^2 + b. \tag{1}$$

where a and b are constants coefficient of communication energy consumption, C_{ij} is the energy consumption of transmitting one package data from the ith sensor to the sink node which is in the jth sink site. Since the real distance of package transmitting of ith sensor is the Euclidean distance between the ith sensor and its parent node in particular MST, d_{ij} represents the actual transmitting distance of the ith SN when sink node is in the jth sink site, i.e.,

$$d_{ij}^2 = (i_x - k_x)^2 + (i_y - k_y^2) \tag{2}$$

where x, y are the coordinate value and k is the parent node of the ith SN in the corresponding MST whose root is jth sink site.

When a WSN starts working, the sink node is initialized to a particular sink site. We assume that each SN has an initial energy of e_0 joule. Then, the sink node will stay at the sink site for certain time interval and move to a new sink site. Every time the sink moves to a new sink site, all SNs need to send a package to the sink node to construct a new MST. Since the model implements the multi-hop manner to transmit the packages, the energy consumption F for the ith SN in setting up the MST is the product of the number of SNs in the ith SN's sub-tree in MST and the energy consumption of the ith SN to transmit one package, i.e.,

$$F_{ij} = C_{ij} * N_{ij}. \tag{3}$$

where F_{ij} is the energy consumption for setting up the new MST for the ith SN when the sink node moves to the j sink site and N_{ij} is the number of SNs in the subtree of node i in MST when the sink node moves to the jth sink site.

The sink will stay at the site at least t_{min}. The sink judge whether it need to move to other sink site every t_{min}, to balance the energy consumption among the nodes. The broadcasting procedure will be omitted if the sink is remained in the original location. The maximum distance that sink can travel from one site to another is denoted as d.

According to [18], a WSN is deemed as dead when there is at least one SN running out of energy. Thus, the lifetime of a WSN can be expressed by

$$T = t_{min} * n, \tag{4}$$

where n is the number of the rounds (each round is a time interval of t_{min}) the WSN can work before it dies. The MSSP requires scheduling the moving paths of the sink properly so as to maximize the network lifetime.

3 Proposed Method

3.1 General Framework

In this paper, we adopt the "Sense-think-act" paradigm to schedule the sink mobility. Based on this paradigm, the sink will sense the environment features at the end of each round. Then, a high-level heuristic (HH) is utilized to determine the next sink site. Finally, the sink will move to the next sink site and stay there for a time interval of t_{min}. The HH is formed by various environment features (or terminals) and numerical functions such as sin and +. Based on the above scheduling mechanism, the problem of maximizing the lifetime of a WSN can be converted to a combinatorial optimization problem: *Given the terminal and function set, find the best HH $\Gamma*$ to determine the next location of the sink so that the network lifetime can be maximized, i.e.,*

$$\Gamma^* = \arg\min_{\Gamma} T(\Gamma), \tag{5}$$

where $T(\Gamma)$ is the network lifetime of the WSN that uses Γ as the HH to schedule the sink.

To search for the best Γ^*, the GP approach is used in the proposed method because GP is quite suitable for finding tree structure heuristic rules and formula

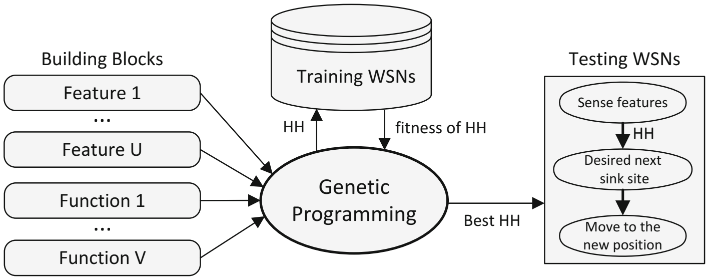

Fig. 2. The general structure of the proposed GP-based framework

[19,22]. Figure 2 illustrates the general structure of the proposed framework. The terminals and functions are defined in advance based on domain knowledge. They are used to build blocks to construct the HH. WSNs with different scales and deploying strategies are generated to evaluate the fitness (or quality) of a HH. That is, the fitness value of a HH is the average network lifetime of the training WSNs where the sink node is scheduled by the HH. Guided by this fitness function, the GP attempts to find a near global optimal HH via an iterative search process. At the end of the GP, the best HH is decoded for practical use.

3.2 Terminal Definition

To facilitate GP constructing promising HH, four low-level heuristics (or terminals) are defined as follows:

– *Minimum nodes energy (λ)* this terminal represents the minimum energy of the nodes covered by the candidate sink site.

$$\lambda = \min \{node\, i's\, energy | \text{SNs covered by the current sink site}\} \tag{6}$$

– *Local global network lifetime (κ)* this terminal represents the network lifetime if sink remain in the candidate sink site till the network dies. The value of κ is computed by:

$$\kappa = \min \left\{ \frac{node\, i's\, energy}{C_{xi}} | \text{SNs covered by the current sink site} \right\} \tag{7}$$

– *Average node energy (μ)* - the average node energy is calculated by:

$$\mu = \frac{\sum_{i=0}^{maxNode} e_i}{maxNode} \tag{8}$$

where $maxNode$ is the number of nodes covered by the candidate sink site and e_i is the ith node's current energy.

– *Average energy consumption of nodes* (ν) -the average energy consumption is calculated by:

$$\nu = \frac{\sum_{i=0}^{maxNode} C_{ix}}{maxNode} \tag{9}$$

where C_{ix} is the energy consumption of transmitting one package from node i to site x.

3.3 Framework Implementation

In this paper, a recent published GP variant named SL-GEP [23] is adopted to search for the optimal HH. In SL-GEP, each chromosome represents a candidate heuristic, which is a fixed-length string that comprises of two parts. The first part is the main program which gives the final output, and the second part contains sub functions to be used in the main program. Both the main program and the sub-function parts can be translated into a mathematical formula by using the Breadth-First-traversal scheme. A typical example of the chromosome of SL-GEP can be expressed as:

$$[ADF_1, *, ADF_2, ADF_2, \kappa, \lambda, \sin, \kappa, \mu, \nu, +, *, b, a, b, -, a, b] \tag{10}$$

Figure 3 illustrates the decoded expression trees of the chromosome.

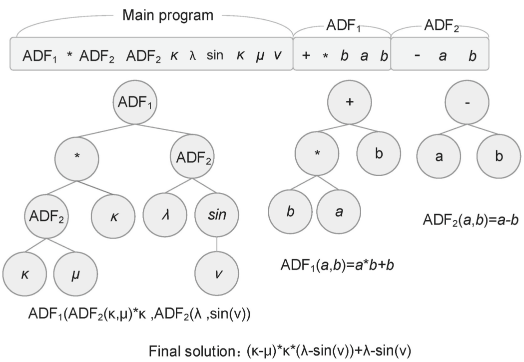

Fig. 3. The illustration of chromosome and tree structure demonstration

Based on the above chromosome representation, the SL-GEP evolves the chromosomes in the following procedures.

The first step is to generate a random initial population. Each chromosome in the population is represented by a vector of symbols:

$$X_i = [x_{i,1}, x_{i,2}, \dots, x_{i,n}] \tag{11}$$

where n is the chromosome length. The value of $x_{i,j}$ is set as a feasible value (e.g., function, terminal, or ADF) randomly.

The second step is mutation which generates a mutant vector for each parent chromosome. In particular, the "DE/current-to-best/1" mutation is adopted to generate the mutant vectors:

$$Y_i = X_i + F \cdot (X_{best} - X_i) + \beta \cdot (X_{r1} - X_{r2}). \tag{12}$$

where X_{best} is the best-so-far chromosome, X_{r1} and X_{r2} are two random individuals selected from the population. X_i, X_{r1}, X_{r2}, and X_{best} are discrete vectors, the numerical operators such as $\cdot$ and $+$ are redefined in SL-GEP to evolve tree-structure solutions. Based on the newly defined operators, each element of Y_i is assigned by mutating the corresponding element in X_i. The mutation probability φ is calculated by the following two sub-steps: sub-step 1: randomly set the values of F and β, i.e., $F = rand(0,1), \beta = rand(0,1)$, where $rand(a,b)$ returns a random value uniformly distributed within $[a,b]$; sub-step 2: For each $x_{i,j}$, a mutation probability is calculated by

$$\varphi = 1 - (1 - F \cdot \psi(x_{best,j}, x_{i,j})) * (1 - \beta \cdot \psi(x_{r1,j}, x_{r2,j})) \tag{13}$$

where $\psi(a,b)$ is defined as

$$\psi(a,b) = \begin{cases} 1, \text{if } a \neq b \\ 0, \text{otherwise} \end{cases} \tag{14}$$

If $y_{i,j} \in$ sub-function, it will be set to a random value, as done in the initialization step. Otherwise, the type of $y_{i,j}$ is set first by considering the frequencies of feasible types appearing in the population. Feasible types that appear more often in the population are more likely to be selected. Once the feasible type of $y_{i,j}$ is determined, $y_{i,j}$ is then set to a random value of the selected feasible type.

In the third step, a crossover is performed to cross each target vector X_i with its mutant vector Y_i to generate a trial vector U_i. i.e.,

$$u_{i,j} = \begin{cases} y_{i,j}, \text{if} \quad rand(0,1) < CR \quad \text{or} \quad j = k \\ x_{i,j}, \text{otherwise} \end{cases} \tag{15}$$

where CR with $CR = rand(0,1)$ is the crossover rate, k is a random integer between 1 and n, $u_{i,j}, y_{i,j}$ and $x_{i,j}$ are the jth variables of U_i, Y_i and X_i, respectively.

Finally, the selection operation selects the fitter solution between each pair of the target and trial vector to form a new population for the next generation, i.e.,

$$X_i = \begin{cases} U_i, \text{if} \quad f\big(U_i\big) > f\big(X_i\big) \\ X_i, \text{otherwise} \end{cases} \tag{16}$$

where $f\big(X\big)$ is the fitness evaluation function which returns the network lifetime.

The second to fourth operations are repeated until the terminated conditions are met.

4 Experiment Studies

4.1 Training and Testing Data

In the simulation studies, we generate WSNs with different features for training and testing. For all WSNs, the SNs are deployed in a rectangle region which is evenly divided into discrete grids. The size of the discrete grids is 6 m and the intersect points of the discrete grids are regarded as candidate sink sites. The initial settings of all WSNs are set to be: $e_0 = 50000\,\mathrm{J}$, $t_{min} = 500\,\mathrm{s}$, $R = 25\,\mathrm{m}$, and $d = 10\,\mathrm{m}$. The SNs are distributed uniformly or with a gaussian distribution. Figure 4 illustrates two examples of the WSNs.

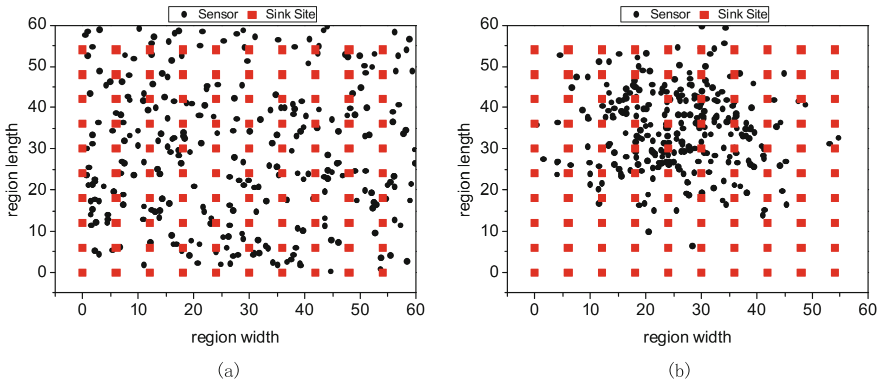

Fig. 4. Typical examples of the training WSNs. (a) a WSN with SNs in uniform distribution. (b) a WSN with SNs in Gaussian distribution.

During the training process, we learn heuristics from three different datasets respectively. The first dataset contains 100 WSNs with uniform distribution of SNs. The second dataset contains 100 WSNs with Gaussian distribution of SNs are used for training. The third dataset contains 100 WSNs with hybrid distribution (i.e., 50 WSNs with uniform distribution and 50 WSNs with Gaussian distribution). Our goal is to investigate which dataset is better for learning general heuristic rule. After the training process, the heuristics learned are test on testing datasets. We generate 60 WSNs with different scales and deploying strategies for testing. Among them, 30 testing WSNs contain SNs with uniform distribution and the remaining 30 WSNs contains SNs with Gaussian distribution. It should be noted that the number of SNs and the size of the regions are different for different testing WSNs.

To investigate the effectiveness of the proposed method, seven heuristics rules are used for comparison. The GMRE is a well-known heuristic proposed in [18]. The RM is a heuristic which selects the next sink site randomly. The four terminals, i.e., κ, λ, μ, and υ are also regarded as heuristic rules for comparison. In addition, we manually design a heuristic by multiplying the four terminals as the last heuristic. As in [18], the network lifetime is considered as the most important metric for performance evaluation.

4.2 Results and Analysis

For each training strategy, the SL-GEP is performed for 100 runs on the training data accordingly. Figure 5 illustrates the evolution of the best network lifetime (in round). It can be observed that for both three cases, the best network lifetime become longer as the evolution goes on, which means that the heuristic rule found by the proposed method become better and better.

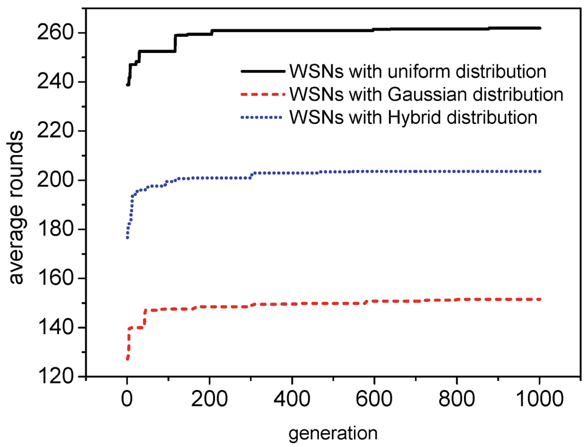

Fig. 5. Fitness value in training

Denote the best heuristics rule learned from the three datasets as $\mathit{\Gamma}_U, \mathit{\Gamma}_G$, and $\mathit{\Gamma}_H$ respectively. Three typical examples of $\mathit{\Gamma}_U, \mathit{\Gamma}_G$, and $\mathit{\Gamma}_H$ are expressed in (17), (18), and (19) respectively.

$$
\begin{aligned}
\mathit{\Gamma}_U =&((((((((\kappa-\lambda)/(\kappa-\lambda))+((\kappa-\lambda)*((\kappa/\mu)*(\kappa*\mu))))\\
&/(((\mu/\mu)*(\mu*\mu))*\mu))*((((\kappa-\lambda)/(\kappa-\lambda))+(\kappa-\lambda)\\
&*((\kappa/\mu)*(\kappa*\mu))))*(((\mu/\mu)*(\mu*\mu))*\mu)))/(((\lambda/\nu)/\\
&(\kappa+\nu))*((\lambda/\nu)*(\kappa+\nu))))*((((((\kappa-\lambda)/(\kappa-\lambda))+((\kappa\\
&-\lambda)*((\kappa/\mu)*(\kappa*\mu))))/(((\mu/\mu)*(\mu*\mu))*\mu))*((((\kappa-\lambda)\\
&/(\kappa-\lambda))+((\kappa-\lambda)*((\kappa/\mu)*(\kappa*\mu))))*(((\mu/\mu)\\
&*(\mu*\mu))*\mu)))*(((\lambda/\nu)/(\kappa+\nu))*((\lambda/\nu)*(\kappa+\nu)))))
\end{aligned} \tag{17}
$$

$$\begin{aligned}\mathit{\Gamma}_G =&((((((((\kappa/\mu) * (\mu * \kappa))/(\nu * \lambda)) * ((\nu * \lambda) * ((\kappa/\mu) * (\mu * \kappa)))) \\ &- ((\kappa * \kappa) + \lambda))/(((\mu * \lambda)/(\mu * \nu)) * ((\mu * \nu) * (\mu * \lambda))))* \\ &((((\mu * \lambda)/(\mu * \nu)) * ((\mu * \nu) * (\mu * \lambda))) * (((((\kappa/\mu) * (\mu * \kappa))/ \\ &(\nu * \lambda)) * ((\nu * \lambda) * ((\kappa/\mu) * (\mu * \kappa)))) - ((\kappa * \kappa) + \lambda))))\end{aligned} \tag{18}$$

$$\begin{aligned}\mathit{\Gamma}_H =&(((\mu - ((\mu - ((\kappa + \mu) * \mu)) + ((\kappa + \mu) * \mu))) + (((\kappa - \\ &((\kappa + \lambda) * \lambda)) + ((\kappa + \lambda) * \lambda))/\lambda)) * (((\kappa - ((\kappa + \lambda) * \lambda)) \\ &+ ((\kappa + \lambda) * \lambda))/\lambda))\end{aligned} \tag{19}$$

Next, we test the performance of the best HH found by using the testing WSNs. Since the number of rounds is in proportion to network lifetime, we use the average rounds to represent the network lifetime. We apply the three heuristics learned from the three datasets to the 60 testing cases (30 WSNs with uniform distribution and 30 WSNs with Gaussian distribution).

Table 1. Comparison results

Function	Case	
	Uniform case	Gaussian case
κ	199.7	87.8
λ	193.3	70.6
μ	193.3	78.8
ν	197.9	74.7
GMRE	193.4	80.7
RM	190.3	71.5
Human designed	193.3	80.7
$\mathit{\Gamma}_U$	**211.2**	**99.7**
$\mathit{\Gamma}_G$	**207.3**	**106.6**
$\mathit{\Gamma}_H$	**210.7**	**103.8**

Table 1 lists the final comparison results. It can be observed that the heuristics generated by SL-GEP can improve the network lifetime by about 8% in uniform distribution and 30% in Gaussian distribution. In addition, the heuristics learned by the SL-GEP can outperform other heuristics designed by human. An interesting discovery is that $\mathit{\Gamma}_U$ performs better than $\mathit{\Gamma}_G$ on uniform case, while $\mathit{\Gamma}_G$ performs better than $\mathit{\Gamma}_U$ on Gaussian case. These results indicate that the heuristic learned from one dataset performs better in testing WSNs that are of the same types.

5 Conclusion

In this paper, we have proposed a GP based approach to maximize the lifetime of WSNs with a mobile sink. In particular, the self-learning gene expression programming is utilized to construct high-level heuristic which can dynamically schedule the moving path of the sink node. To evaluate the proposed approach, empirical study on WSNs with different scales and deployment strategies are conducted. The superior performance obtained against several other approaches confirmed the efficacy of our proposed method. As for future work, an interesting research direction to extend the proposed framework to optimize multiple objectives, such as the path length of the sink and the readability (or complexity) of the solution.

Acknowledgment. This work is partially supported under the National Natural Science Foundation of China (Grant Nos. 61602181, 61603064), Fundamental Research Funds for the Central Universities (Grant No. 2017ZD053), Frontier Interdisciplinary Research Fund for the Central Universities (Grant No. 106112017CDJQJ188828).

References

1. Bouyahi, M., Ezzedine, T.: Design of smart Bridge based on WSN for efficient measuring of temperature, strain and humidity. In: 2016 4th International Conference on Control Engineering Information Technology (CEIT), pp. 1–5 (2016)
2. Boubrima, A., Bechkit, W., Rivano, H.: Optimal WSN deployment models for air pollution monitoring. IEEE Trans. Wirel. Commun. **16**(5), 2723–2735 (2017)
3. Ren, G.L., Khairi, N.A.B.F., Ismail, W.: Design and implementation of environmental monitoring using RFID and WSN platform. In:2016 IEEE Asia-Pacific Conference on Applied Electromagnetics (APACE), pp. 328–333 (2016)
4. Lu, M., Zhao, X., Huang, Y.: Fast localization for emergency monitoring and rescue in disaster scenarios based on WSN. In: 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp. 1–6. (2016). doi:10.1109/ICARCV.2016.7838790
5. Agemura, S., Katayama, K., Ohsaki, H.: On the effect of wireless communication range heterogeneity on WSN performance. In: International Conference on Information Networking (ICOIN), pp. 35–40 (2017)
6. Arunachalam, B., Arjun, D., Prahlada, R.B., Pasupuleti, H., Dwarakanath, V.: Sensing service framework for climate alert system using WSN-cloud infrastructure. In: 2015 9th International Conference on Sensing Technology (ICST), pp. 671–676 (2015)
7. Alaiad, A., Zhou, L.: Patients' adoption of WSN-based smart home healthcare systems: an integrated model of facilitators and barriers. IEEE Trans. Prof. Commun. **60**(1), 4–23 (2017)
8. Alvi, A.N., Bouk, S.H., Ahmed, S.H., Yaqub, M.A., Sarkar, M., Song, H.: BEST-MAC: bitmap-assisted efficient and scalable TDMA-based WSN MAC protocol for smart cities. IEEE Access **4**, 312–322 (2016)
9. Ye, Y., Luo, H., Cheng, J., Lu, S., Zhang, L.: A two-tier data dissemination model for large-scale wireless sensor networks. In: Proceedings of the 8th Annual International Conference on Mobile Computing and Networking, pp. 148–159. ACM (2002)
10. Lin, C.-J., Chou, P.-L., Chou, C.-F.: HCDD: hierarchical cluster-based data dissemination in wireless sensor networks with mobile sink. In: Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing, pp. 1189–1194. ACM (2006)

11. Jea, D., Somasundara, A., Srivastava, M.: Multiple controlled mobile elements (data mules) for data collection in sensor networks. In: Prasanna, V.K., Iyengar, S.S., Spirakis, P.G., Welsh, M. (eds.) DCOSS 2005. LNCS, vol. 3560, pp. 244–257. Springer, Heidelberg (2005). doi:10.1007/11502593_20
12. Kushal, B.Y., Chitra, M.: Cluster based routing protocol to prolong network lifetime through mobile sink in WSN. In: 2016 IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT), pp. 1287–1291 (2016)
13. Ren, J., Zhang, Y., Zhang, K., Liu, A., Chen, J., Shen, X.S.: Lifetime and energy hole evolution analysis in data-gathering wireless sensor networks. IEEE Trans. Ind. Inform. **12**(2), 788–800 (2016)
14. Wang, Z.M., Basagni, S., Melachrinoudis, E., Petrioli, C.: Exploiting sink mobility for maximizing sensor networks lifetime. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences, HICSS 2005, p. 287a. IEEE (2005)
15. Yun, Y., Xia, Y.: Maximizing the lifetime of wireless sensor networks with mobile sink in delay-tolerant applications. IEEE Trans. Mob. Comput. **9**(9), 1308–1318 (2010)
16. Shi, Y., Hou, Y.T.: Theoretical results on base station movement problem for sensor network. In: The 27th Conference on Computer Communications INFOCOM 2008, pp. 1–5. IEEE (2008)
17. Zhong, J., Zhang, J.: Ant colony optimization algorithm for lifetime maximization in wireless sensor network with mobile sink. In: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, pp. 1199–1204. ACM (2012)
18. Basagni, S., Carosi, A., Melachrinoudis, E., Petrioli, C., Wang, Z.M.: Controlled sink mobility for prolonging wireless sensor networks lifetime. Wirel. Netw. **14**(6), 831–858 (2008)
19. Zhong, J., Cai, W., Lees, M., Luo, L.: Automatic model construction for the behavior of human crowds. Appl. Soft Comput. **56**, 368–378 (2017)
20. Zhong, J., Feng, L., Ong, Y.-S.: Gene expression programming: a survey. IEEE Comput. Intell. Mag. **12**(3), 54–72 (2017)
21. Tay, J.C., Ho, N.B.: Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems. Comput. Ind. Eng. **54**(3), 453–473 (2008)
22. Xiao, Q., Zhong, J., Chen, W.N., Zhan, Z.H., Zhang, J.: Indicator-based multi-objective genetic programming for workflow scheduling problem. In: 2017 Genetic and Evolutionary Computation Conference Companion (GECCO), pp. 217–218 (2017)
23. Zhong, J., Ong, Y.S., Cai, W.: Self-learning gene expression programming. IEEE Trans. Evol. Comput. **20**(1), 65–80 (2016)
24. Bhatti, R., Kaur, G.: Virtual grid based energy efficient mobile sink routing algorithm for WSN. In: 2017 11th International Conference on Intelligent Systems and Control (ISCO), pp. 30–33 (2017)
25. Heinzelman, W.R., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, pp. 1–10 (2000)