World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:9, No:1, 2015

Harmony Search-based K-Coverage Enhancement in Wireless Sensor Networks

Shaimaa M. Mohamed, Haitham S. Hamza, Imane A. Saroit

Abstract—Many wireless sensor network applications require K-coverage of the monitored area. In this paper, we propose a scalable harmony search based algorithm in terms of execution time, K-Coverage Enhancement Algorithm (KCEA), it attempts to enhance initial coverage, and achieve the required K-coverage degree for a specific application efficiently. Simulation results show that the proposed algorithm achieves coverage improvement of 5.34% compared to K-Coverage Rate Deployment (K-CRD), which achieves 1.31% when deploying one additional sensor. Moreover, the proposed algorithm is more time efficient.

Keywords—Wireless Sensor Networks (WSN), Harmony Search Algorithms, K-Coverage, Mobile WSN.

I. Introduction

WIRELESS sensor network, (WSN), has been deployed widely in various applications, such as structural [1], [2], environmental [3]–[5], habitat monitoring [6]–[9], military surveillance [10], and intrusion detection [11]. A WSN typically consists of a large number of sensor nodes, and one or more sink nodes. Those nodes are usually deployed to monitor certain event or phenomenon [12]. One of the main challenges in WSN, and a key to the successful operation of several applications is coverage [8].

WSN applications differ greatly in their coverage requirements. Some applications do not need coverage, such as monitoring human physiology data, while others require each point to be covered by only one sensor, such as habitat monitoring. Whereas, other applications require more than one sensor to monitor each point, such as surveillance, and intrusion detection applications.

There are generally two methods for deploying sensor nodes: 1) deterministic deployment, where node locations are calculated prior to deployment. 2) random deployment, where it is more practical in some environments, such as in [13], [14]. Using random deployment [15]–[18] may not achieve the required coverage; therefore, algorithms are needed after initial random deployment to ensure that the required coverage degree is achieved.

Dynamic deployment [18]–[21] is one solution to enhance initial random coverage. It ensures full coverage by redistributing nodes after initial random deployment.

In this paper, we consider the case where there are additional sensor nodes available after initial random deployment, and rather than redistributing the deployed nodes, we calculate optimal positions for placing more nodes, so that the coverage degree required by the application is achieved. We propose use a harmony search based algorithm for achieving the required K-coverage, so the proposed algorithm is named K-Coverage Enhancement Algorithm (KCEA).

Simulation results show that KCEA outperforms K-Coverage Rate Deployment (K-CRD) [22], in terms of both the coverage improvement, and execution time. In deployments with initially 30 sensors, KCEA achieves 28.42%, while K-CRD achieves 19.8% coverage improvement, when deploying 6 additional sensors. Moreover, KCEA produces these improvements in $14s$; whereas, K-CRD takes $75s$. In deployments with initially 60 sensors, KCEA achieves 13.81% overage improvement with $4.8s$ execution time, while K-CRD achieves 10.2% coverage improvement with $45.8s$ execution time. Also, simulations show that the proposed solution is scalable (in terms of execution time); as the number of additional nodes increase, the required coverage degree K increases, and the maximum tolerable evaluation error (MTEE) [22] decreases.

The remainder of the paper is organized as follows. Related work is presented is Section II. The proposed K-Coverage Enhancement Algorithm (KCEA) is presented in Section III. Simulation experiments are presented in Section IV. Finally, concluding remarks are presented in Section V.

II. Related Work

A number of literature considered the coverage problem in WSN, which is concerned with whether every point in the deployment area is monitored by at least one of the sensor nodes [19], [23]–[25]. A generalization to this problem is K-coverage [21], [22], [26], [27], where every point in the deployment area should be within the sensing radii of K-distinct sensors. K refers to the coverage degree (or level).

In [22], the authors consider two problems: 1) coverage determination and 2) coverage deployment. A K-Coverage Rate Evaluation (K-CRE) algorithm is proposed to determine the coverage degree of the monitored area. The algorithm uses non-uniform grids, and calculates coverage information of each grid. Grids with uncertain coverage information are further divided into sub-grids. The sud-griding process is repeated until the ratio of uncertainly covered area relative to whole monitored area (Maximum Evaluation Error) is greater than the Maximum Tolerable Evaluation Error (MTEE) permitted for a target application. In our proposed algorithm, K-CRE is used for evaluating the coverage of the monitored area.

In addition, the authors present a K-Coverage Rate Deployment (K-CRD) algorithm, that can be used if the required K-coverage is not achieved, and there are additional

Shaimaa M. Mohamed, Haitham S. Hamza and Imane A. Saroit are with the Department of Information Technology, Faculty of Computers & Information, Cairo (e-mail: s.mosaad@fci-cu.edu.eg).

sensor nodes available for deployment. The algorithm uses deployment regions of the grids to determine optimal positions for node placement. The deployment region is the area where if a sensor is deployed, it will fully cover the grid. In an attempt to reduce the computational cost, K-CRD1 algorithm is proposed, in which only α-grids rather than all the grids are considered.

The work in [26] investigates the use of Ant Colony Optimization (ACO) to find the optimal positions for node deployments. The authors propose EasiDesign, an improved ACO algorithm that finds positions that minimize cost, achieves both connectivity, and K-coverage.

The work in [28] studies coverage in hybrid WSN. The authors assume that there are a number of static nodes deployed randomly, and determine the optimal locations for placing m mobile nodes using Artificial Bee Colony (ABC) algorithm.

III. The Proposed Algorithm

A. Original Harmony Search (HS) Algorithm

Harmony search (HS) is an optimization algorithm, which is based on the experiences of Jazz musicians [29]. HS has three main phases: Parameter Initialization, Improvising, and updating. The algorithm starts with an initial random population (harmonic) stored in Harmony Memory (HM), for which an optimization function is calculated for each of its harmonic (solution vector). HS depends on three operations to explore the search space, and generate a new harmonic: harmony memory consideration, pitch adjusting, and randomization. Memory consideration carries the best solutions to the new population. Pitch adjustment adjusts a solution to a neighboring value, while randomization explores new solutions. HS generates only one new harmonic, which replaces the worst one in the older population.

Since its development, a number of variants has been proposed [30]–[33]. Local-best harmony search algorithm with dynamic subpopulations (DLHS) is one of the HS variants (Fig. 1), which attempts to enhance the parameter initialization phase, and solve continuous optimization problems [34]. It uses an adaptive parameter selection strategy, and divides the HM into a number of small-sized sub-HMs, where improvisation is performed independently on each sub-HM. The sub-HMs are regrouped frequently in order to retain population diversity, and to exchange information among the solutions (The dotted parts in Fig. 1).

1. Problem and Parameter Initialization

During parameter initialization step, the following parameters are initialized:

- Harmony memory size (HMS), the population size.
- Number of decision variables (N).
- The number of iterations (NI).
- The harmony memory (HM) is populated randomly with solution vectors.

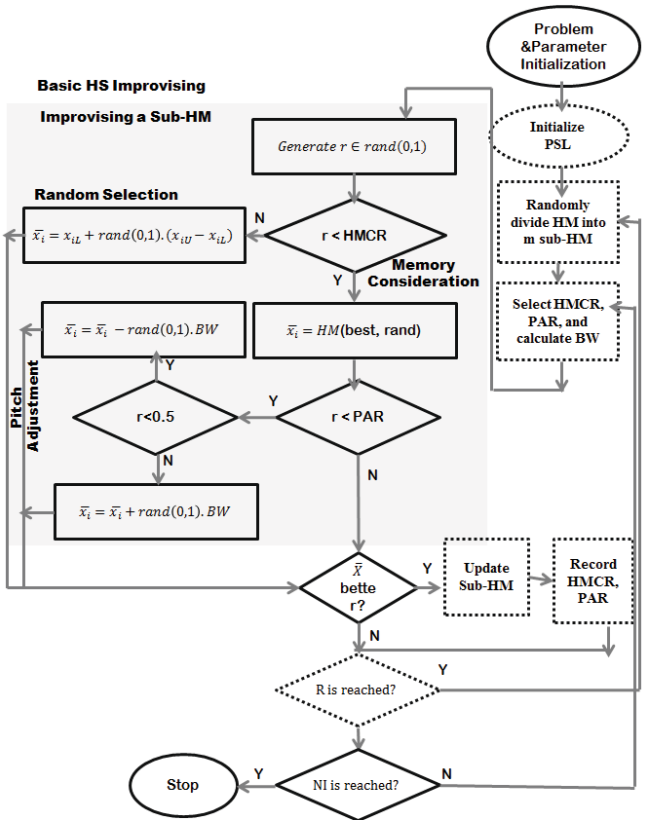


Fig. 1 DLHS Flowchart

2. PSL initialization

The DLHS algorithm uses an adaptive learning mechanism to determine the best values of $HMCR$, and PAR in order to achieve both good exploration, and exploitation. A parameter set list (PSL) is initialized randomly, containing random values of $HMCR$, and PAR, such that $HMCR \in [0.9, 1]$, and $PAR \in [0, 1]$ as in [34]. At the start of each iteration, the values of $HMCR$, and PAR are selected from the PSL list. $HMCR$, and PAR values, that generate a harmony better than all those stored in the sub-HM are added to a new list called a winning parameter set list ($WPSL$). Once all the $HMCR$, and PAR values in the PSL are exhausted, it is refilled from the $WPSL$ with probability 0.75, or randomly with probability 0.25 [34]. Similarly, the BW value changes dynamically such that, it decreases with the number of iterations.

3. Improvising

During the improvising phase, the HM is randomly divided into m subpopulations. The $HMCR$, PAR, and BW values are selected from the PSL list. Then, a new solution vector $\overline{X}$ is generated using one of three operations: random selection, memory consideration or pitch adjustment.

The new harmony vector $\overline{X}$ replaces the worst harmony vector in its sub-HM, if it has better objective function. The $HMCR$, and PAR values are added to $WPSL$ list. Every R iterations, the subpopulations are regrouped again to retain the population diversity, and to exchange information among the solutions. The process is repeated until the maximum number

of iterations (NI) is reached.

B. K-Coverage Enhancement Algorithm (KCEA)

In some applications, K-coverage rather than 1-coverage is required. An area has a coverage degree (level) of K, or called K-covered, if every point in the area is within sensing radii of K-distinct sensors. Assuming a disk binary sensing model, a point $p = (x, y)$ is covered by a sensor s if the distance between them is less than or equal to r_s as in (1).

$$C_s(p) = \begin{cases} 1 & if\ d(s,p) \leq r_s \\ 0 & o.w. \end{cases} \quad (1)$$

where,
$d(s,p)$ is the distance between a sensor node s and a point p.
r_s is the sensing radius of the nodes.

If a point can be covered by more than one sensor, the total coverage of a point is $C(p) = \sum_{i=1}^{k} C_{Si}(p)$ [35]; therefore, K-coverage can be achieved if:

$$C(p) = k \ \ \forall p \in A \quad (2)$$

Using initial random deployment, neither 1-coverage nor K-coverage can be guaranteed. In these cases, if there are additional sensor nodes available after initial deployment, they can be used to achieve the required coverage degree. The main problem is how to determine the positions of these nodes efficiently, in order to achieve the required coverage degree, and to minimize the computational cost. We propose a K-Coverage Enhancement Algorithm (KCEA), which attempts to improve the required coverage degree in such cases. The algorithm uses the DLHS to find the optimal places for deploying additional nodes efficiently.

KCEA starts with dividing the deployment area into non-uniform grids using K-CRE algorithm [22]. Then, the coverage status for each grid is determined. Based on this information, a number of grids are identified as candidate grids. Candidate grids are those covered by less than K sensor nodes. Those grids are, then ordered according to their area, so that large grids have higher priority for sensor placement than smaller ones.

The idea of our algorithm (Algorithm 1) is to search heuristically using DLHS for the optimal position to place an additional sensor within each candidate grid, such that K-coverage is improved. Therefore, in KCEA algorithm, HM contains harmonies equal to the number of candidate grids (3). Each harmony represents a possible placement for one, or more additional sensors depending on the required coverage degree K. Each decision variable represents (x, y) coordinates of the sensor node. Although, the positions of the sensor nodes are chosen randomly, they must be within the boundaries of one of the candidate grids (4). At the end of each iteration, K-coverage is evaluated (5) and the new harmony is placed in the HM, if the achieved K-coverage is better than the worst one in the HM.

Algorithm 1 K-Coverage Enhancement Algorithm (KCEA)

Identify the candidate grids as in KCRD algorithm
Set HMS equals to number of candidate grids
Set N equals to number of additional sensors available
Initialize PSL
procedure IMPROVISE
  while $n \leq NI$ **do**
    if $PSL \neq \phi$ **then**
      Select a set of HMCR and PAR from PSL
    else
      Refill PSL
    if $n \leq (NI/2)$ **then**
      $BW(m) = BW_{max} - \frac{BW_{max} - BW_{min}}{NI}.\ 2n$
    else
      $BW(m) = BW_{min}$
    while $i \leq N$ **do**
      if $rand() \leq HMCR$ **then**
        $\overline{x}_i = x_i^{best}, best \in (1, 2, 3, ...HMS)$ ▷ Memory Consideration
        $\overline{y}_i = y_i^{best}, best \in (1, 2, 3, ...HMS)$
        if $rand() \leq PAR$ **then**
          $\overline{x}_i = \overline{x}_i rand() \pm\ .\ BW(m)$ ▷ Pitch adjustment
          $\overline{y}_i = \overline{y}_i rand() \pm\ .\ BW(m)$
      else
        $\overline{x}_i = g_x^L(j) + rand().(g_x^U(j) - g_x^L(j), j \in (1, 2, 3, ...HMS)$ ▷ Random selection
        $\overline{y}_i = g_y^L(j) + rand().(g_y^U(j) - g_y^L(j), j \in (1, 2, 3, ...HMS)$
    Calculate $F(\overline{X})$
    if $F(\overline{X}) > F(X_{worst})$ **then**
      $X_{worst} = \overline{X}$
      Record the HMCR and PAR into WPSL

$$HM = \begin{pmatrix} (x_1^1, y_1^1) \cdots & (x_1^N, y_1^N) \mid F(X_1) \\ (x_2^1, y_2^1) \cdots & (x_2^N, y_2^N) \mid F(X_2) \\ \vdots & \vdots \\ (x_{HMS}^1, y_{HMS}^1) & (x_{HMS}^N, y_{HMS}^N) \mid F(X_{HMS}) \end{pmatrix} \quad (3)$$

$$x_i^j = g_x^L(j) + rand(0,1).(g_x^U(j) - g_x^L(j))$$
$$y_i^j = g_y^L(j) + rand(0,1).(g_y^U(j) - g_y^L(j)) \quad (4)$$

where,
HMS equals the number of candidate grids.
X represents additional sensor possible position within the candidate grid.
N is the number of decision variables and equals the number of additional sensors available.
$F(X)$ is the objective function of decision variable X

(5).
g_j^L, g_j^U is the area of candidate grid j.

$$Max.(F) = C_K \tag{5}$$

where,
F is the objective function to be maximized.
C_K is K-coverage of the the deployment area.

The algorithm parameters are listed in Table I. Since placing a sensor in one of the candidate grids (which have the largest grid weight GW) is an efficient way to improve coverage [22], in the proposed algorithm, HMS is set equal to the number of candidate grids N_{g_β}. In order to measure the performance of the algorithm, the number of additional sensors (N) is increased from 1 to 48. The number of iterations NI is chosen to minimize the execution time and maximize the coverage improvement. The rest of the parameters are set as in [34].

TABLE I
K-COVERAGE ENHANCEMENT ALGORITHM (KCEA) PARAMETERS

Parameter	Value
HMS	N_{g_β}
N	varies from 1 to 48
NI	250
R	50
m	3
$BWmin$	0.0001
$BWmax$	$UB - LB/200$
PSL length	200

IV. Simulation Experiments

A. Simulation Setup Parameters

To evaluate the proposed algorithm, a simulator is implemented using Java language. All the experiments were evaluated on a Intel Core 2 Duo 2.1 GHz processor with 2 MB cache memory, and 2 GB RAM. The simulator considers a WSN initially consisting of a number of sensors n, that are randomly deployed in a $2D$ rectangular area A. Sensors are homogeneous, using a binary disk sensing model with radius r_s. For simplicity, network channel is assumed to be error-free and collision-free. Table II lists the simulation parameters. Since the results of the proposed algorithm (KCEA) are compared to the results of the K-CRD1 algorithm, the same values of the parameters are used as in [22].

TABLE II
SIMULATION PARAMETERS

Parameter	Value
n	30, 60 and 90
A	100 m x 100 m
r_s	10 m
K	$K = 1, 2$ and 3
$MTEE$	0.04, 0.02, and 0.01
Number of runs in experiment set	20 runs

B. Performance Metrics

In order to evaluate the performance of our proposed dynamic deployment algorithm, the following metrics are used:

1) **Coverage Improvement Percentage** ($C\%$)**:** coverage rate is to the ratio of covered area relative to the total monitored area. Coverage improvement percentage is calculated using (6). In order to calculate K-coverage, a modified version of the K-CRE scheme [22] is used. The main problem with the K-CRE scheme is its execution time, especially with smaller MTEE values. So K-CRE is used once to determine the initial coverage, then only the area of the grids which are covered by the new deployed sensor nodes are added to this initial coverage. K-CRE is used one more time after the KCEA terminates, to evaluate the coverage of the final deployment.

$$C\% = (\frac{C_{final} - C_{initial}}{C_{initial}})\text{x } 100 \tag{6}$$

2) **Execution Time** (t)**:** execution time expressed in seconds is used to evaluate the computational speed of our proposed algorithm.

Simulations are conducted to evaluate the performance of the proposed algorithm for the following scenarios: 1) various number of additional sensors (N), 2) various coverage degree (K), and 3) various MTEE.

C. K-CRE Simulation Results

In order to evaluate the performance of our proposed algorithm, we have implemented both the K-CRE, and K-CRD1 schemes presented in [22]. In evaluating the performance of K-CRE scheme, the number of grids was used as the basic metric for measuring the computational cost of the algorithm. We have obtained the same number of grids using our simulator as those reported in [22]. Fig. 2 shows the number of grids calculated using our simulator for $K = 1$.

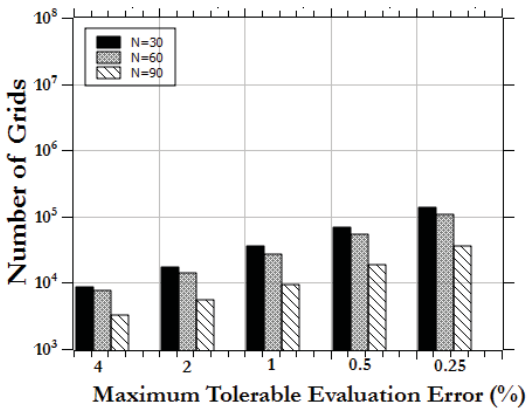
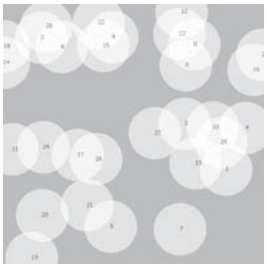


Fig. 2 Number of grids versus MTEE for $k = 1$

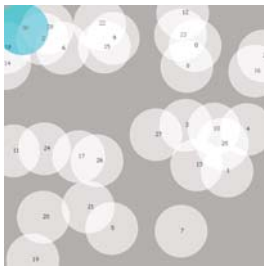
In addition, we have obtained coverage improvement percentages using our implementation of the K-CRD1 scheme similar to the ones reported in [22].

D. Simulation Results of Deploying N Sensors

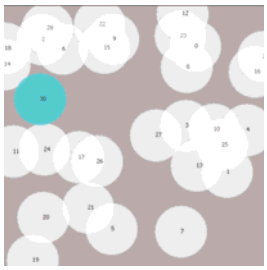
An example of deploying an additional sensor node, after initial random deployment is shown in Fig. 3. Fig. 3(a) shows the initial random deployment, the deployment has a coverage rate of 54.3%. Fig. 3(b) shows the deployment after adding one sensor node using K-CRD1 algorithm, the deployment has a coverage rate of 55.01% with coverage improvement of 1.31%. Fig. 3(c) shows the deployment after adding one sensor node using our proposed KCEA algorithm, the deployment has a coverage rate of 57.2% with coverage improvement of 5.34%. Clearly, our proposed algorithm achieves better coverage improvement than the one achieved by K-CRD1.



(a) Initial Deployment



(b) Deployment after using K-CRD1



(c) Deployment after using KCEA

Fig. 3 Example of Deployment of an Additional Sensor

Table III shows both the coverage improvement percentages, and the execution time in seconds for KCEA, K-CRD (considering all the candidate grids), and K-CRD1 (considering only α candidate grids with $\alpha = 3$). Increasing the number of candidate grids used in K-CRD algorithm calculations, increases the execution time greatly.

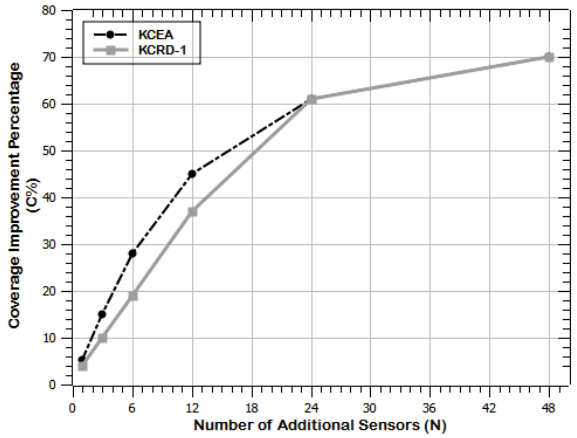
TABLE III
EXECUTION TIME VS. COVERAGE IMPROVEMNT PERCENTAGE $K = 1$ AND $N = 30$

MTEE(%)	Execution Time(s)			Coverage Improvement (%)		
	KCEA	K-CRD1	K-CRD	KCEA	K-CRD-1	K-CRD
4	3.01	2.09	679	5.32	3.77	3.01
2	2.8	9.2	2208	5.42	3.97	3.07
1	2.8	33.1	8866	5.4	3.4	3.1

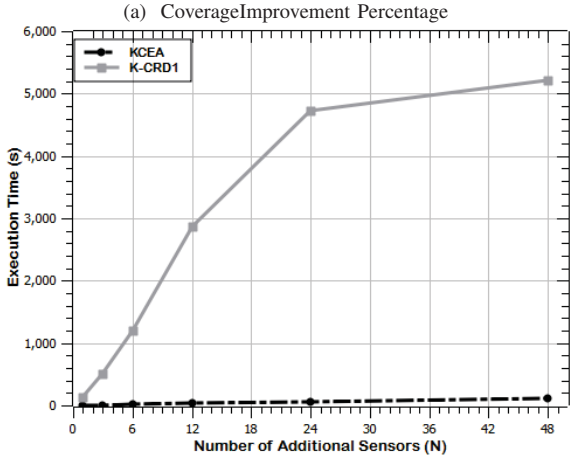
The proposed KCEA outperforms both K-CRD, and K-CRD1 in the achieved coverage improvement. In addition, it has less execution time except for the case when MTEE=0.04.

Fig. 4(a) shows the coverage improvement percentage for $n = 30$, $MTEE = 0.005$, and $K = 1$, while Fig. 4(b) shows the execution time for the same values. KCEA achieves higher coverage improvement percentages compared to K-CRD1 scheme, and with less execution time. This is mainly because the proposed algorithm (KCEA) searches heuristically in all the candidate grids, while K-CRD1 considers only the deployment regions of α candidate grids. In addition, K-CRD1 mainly searches for the intersection between the deployment region of the candidate grids, this may result in a position where the sensing region of the sensor overlap with other sensors or not. KCEA; on the other hand, searches in the whole grid area for the position that maximizes coverage. Moreover, KCEA searches efficiently all of the candidate grids, since it is based on DLHS algorithm.

While considering all the candidate grids deployment regions may improve K-CRD1 results, it will increase the execution time dramatically. Moreover, increasing the number of additional sensors N, increases the execution time of KCEA linearly, while the execution time of the K-CRD-1 increases exponentially.



(a) CoverageImprovement Percentage



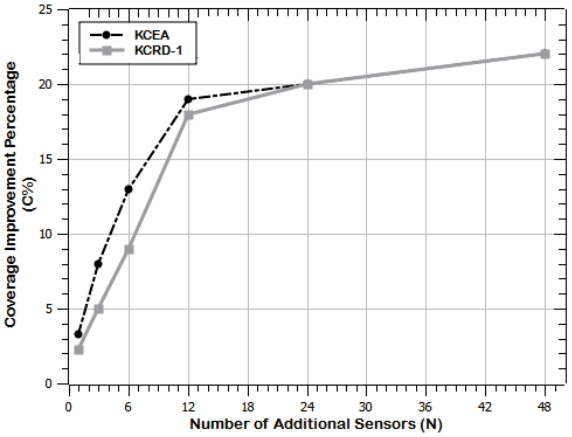
(b) Execution Time

Fig. 4 Coverage improvement percentage, and Execution time for $K = 1$, $n = 30$, and $MTEE = 0.005$

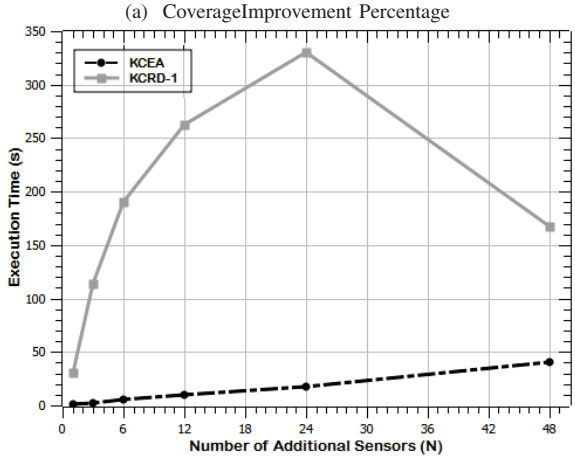
The optimal number of sensors needed to achieve full coverage using a deterministic pattern (the strip-based pattern [36]) can be calculated as given in [37]. Using a deployment area of (100 x 100 m^2), and $r_s = 10m$, the minimum theoretical number of sensor to achieve 1-coverage is 40 sensors. Consequently, both algorithms have similar coverage improvement percentages, when the number of additional sensors is more than 24. Since, most of the deployment area

is covered, and there are less candidate grids to be considered.

Fig. 5 shows the coverage improvement percentage, and execution time for $n = 60$, $MTEE = 0.01$, and $K = 1$. The fig. shows similar behavior as Fig. 4, as KCEA achieves better coverage improvement, and less execution time compared to K-CRD1.



(a) CoverageImprovement Percentage



(b) Execution Time

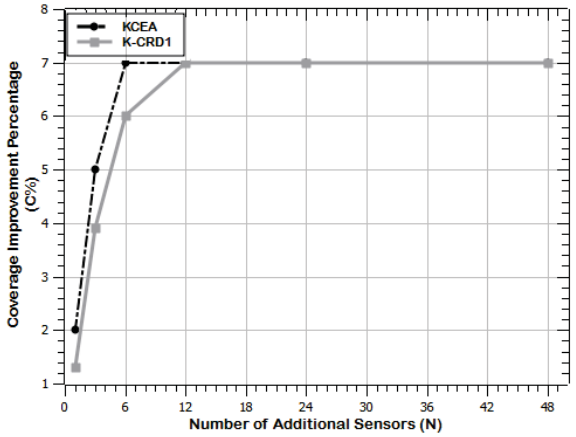
Fig. 5 Coverage improvement percentage, and Execution time for $K = 1$, $n = 60$, and $MTEE = 0.001$

As the number of original deployed sensor nodes n increases, the coverage improvement gap between the two algorithms decreases, because most of the deployment area is already 1-fully covered (Fig. 4, 5, and 6).

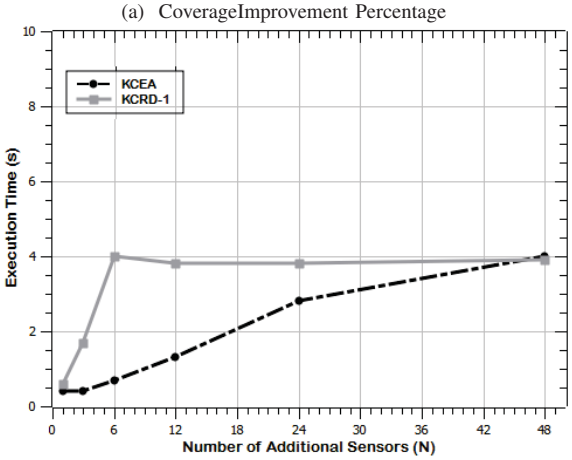
Fig. 4 (b), 5 (b), and 6 (b) show that our proopsed algorithm is scalable in terms of the execution time. While increasing the number of additional sensors N affects the execution time of K-CRD1 exponentially, our proposed algorithm scales linearly.

E. Simulation Results of Achieving $K > 1$

Fig. 7 and 9 show the coverage improvement percentage for $K = 2$ and $n = 90$, $K = 3$ and $n = 30$. While, Fig. 8 and 10 show the execution time for $K = 2$ and $n = 90$, $K = 3$ and $n = 30$. Although, K-CRD1 achieves slightly better coverage improvement percentages, KCEA has much less execution time. Since KCEA search heuristically for the optimal position to place additional the sensor.



(a) CoverageImprovement Percentage



(b) Execution Time

Fig. 6 Coverage improvement percentage, and Execution time for $K = 1$, $n = 90$, and $MTEE = 0.002$

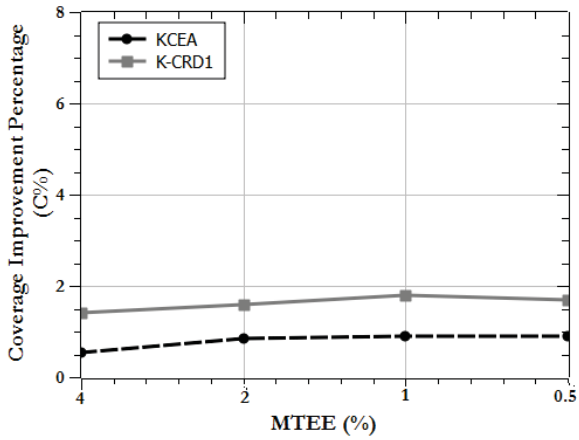


Fig. 7 Coverage improvement percentage for $K = 2$, $n = 90$, and $N = 1$

Different applications may require various Maximum Tolerable Evaluation Error (MTEE). Decreasing the MTEE, increases the execution time of K-CRD1 scheme; conversely, it has no effect on the execution time of the KCEA algorithm.

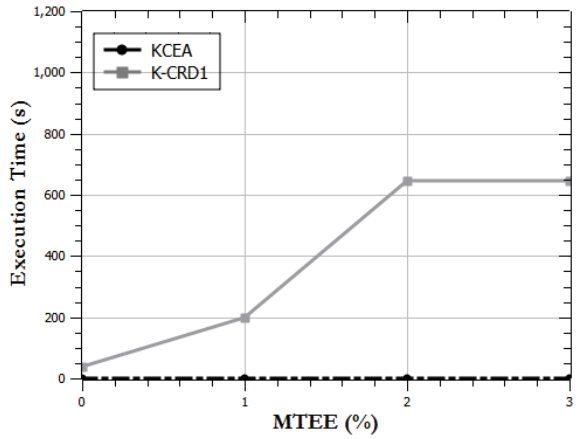


Fig. 8 Execution time for $K = 2$, $n = 90$, and $N = 1$

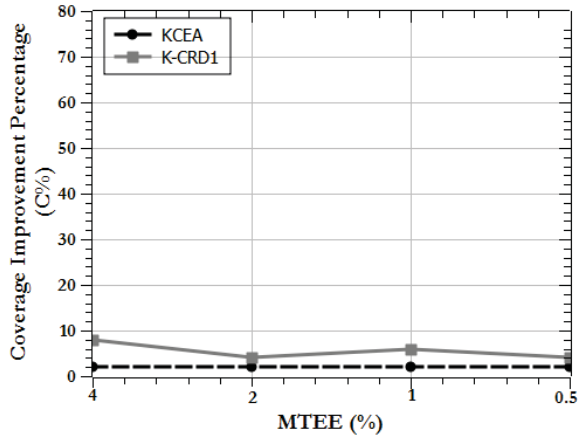


Fig. 9 Coverage improvement percentage for $K = 3$, $n = 30$, and $N = 1$

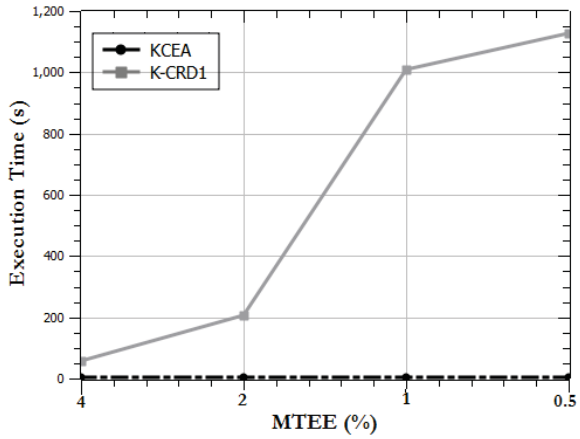


Fig. 10 Execution time for $K = 3$, $n = 30$, and $N = 1$

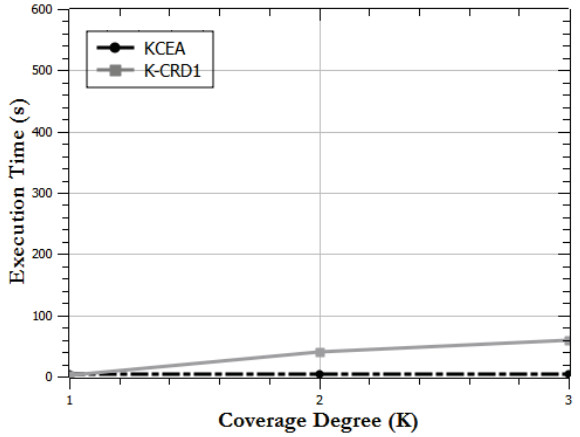


Fig. 11 Execution time versus value of K for $MTEE = 0.04$, $N = 1$, and $n = 30$

F. Execution Time

Execution time is used as a metric for measuring computational cost, and in evaluating the scalability of the algorithm. Figs. 11, 12, and 13 show the execution time in seconds for $n = 30, 60, 90$, respectively, and $N = 1$, and $MTEE = 0.04$. Increasing the coverage degree K, increases the execution time of K-CRD1 scheme greatly, especially for large n. Thus, limiting the scalability of the scheme. Whereas, increasing the coverage degree K has a small effect on the execution time of the proposed algorithm.

Increasing the coverage degree K, the number of initially deployed sensors n, or the number of additional sensors N results in longer execution time. Descreasing the MTEE also results in longer execution time. However, KCEA has much less execution time compared to K-CRD1, when varying any of the previous factors. This mainly because KCEA is based on HS algorithm; therefore, requires less computations than K-CRD1.

V. Conclusion

In this paper, a K-coverage enhancement algorithm, KCEA is proposed. It attempts to efficiently place additional sensors in order to achieve the coverage degree required by the application. KCEA achieves better coverage improvement percentages compared to K-CRD1, with much less execution time. Simulation results show the scalability of our proposed algorithm in terms of execution time. This is due to the algorithm ability to search heuristically in all the candidate grids for the optimal position to place the additional sensor

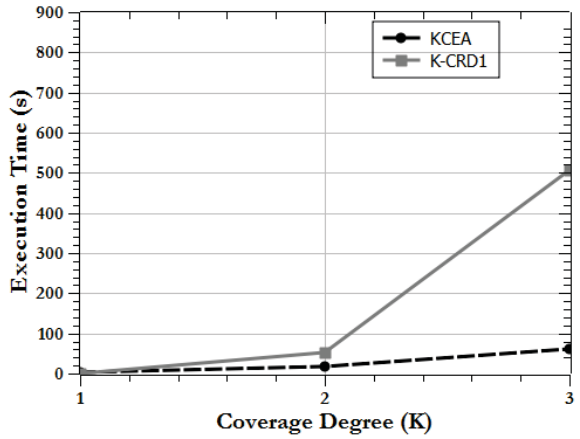


Fig. 12 Execution time versus value of K for $MTEE = 0.04$, $N = 1$, and $n = 60$

nodes. Our ongoing work investigates how to explore the ability of continuous optimization of the DLHS algorithm to continuously maintain coverage as the topology changes due to node failure.

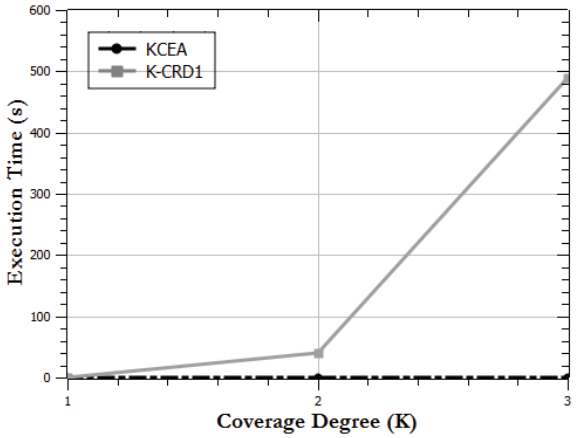


Fig. 13 Execution time versus value of K for $MTEE = 0.04$, $N = 1$, and $n = 90$

REFERENCES

[1] J. Paek, K. Chintalapudi, R. Govindan, J. Caffrey, and S. Masri, "A wireless sensor network for structural health monitoring: Performance and experience," in *Embedded Networked Sensors, 2005. EmNetS-II. The Second IEEE Workshop on*, pp. 1–10, 2005.

[2] G. Anastasi, G. Lo Re, and M. Ortolani, "Wsns for structural health monitoring of historical buildings," in *Human System Interactions, 2009. HSI '09. 2nd Conference on*, pp. 574–579, 2009.

[3] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, SenSys '05, (New York, NY, USA), pp. 51–63, ACM, 2005.

[4] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh, "Monitoring volcanic eruptions with a wireless sensor network," in *Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on*, pp. 108–120, 2005.

[5] H. Y. Jeonghwan Hwang, Changsun Shin, "Study on an agricultural environment monitoring server system using wireless sensor networks," 2010.

[6] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet," *SIGARCH Comput. Archit. News*, vol. 30, pp. 96–107, Oct. 2002.

[7] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, WSNA '02, (New York, NY, USA), pp. 88–97, ACM, 2002.

[8] W. Hu, V. N. Tran, N. Bulusu, C. T. Chou, S. Jha, and A. Taylor, "The design and evaluation of a hybrid sensor network for cane-toad monitoring," in *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pp. 503–508, 2005.

[9] A.-K. Othman, K. M. Lee, H. Zen, W. Zainal, and M. F. M. Sabri, "Wireless sensor networks for swift bird farms monitoring," in *Ultra Modern Telecommunications Workshops, 2009. ICUMT '09. International Conference on*, pp. 1–7, 2009.

[10] M. Hussain, P. Khan, and K. kyung Sup, "Wsn research activities for military application," in *Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on*, vol. 01, pp. 271–274, 2009.

[11] M. Khanafer, M. Guennoun, and H. Mouftah, "Intrusion detection system for wsn-based intelligent transportation systems," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pp. 1–6, Dec 2010.

[12] *Wireless Sensor Networks: Technology, Protocols, and Applications*. Wiley, 2007.

[13] M. S. K. Arash Nikdel and S. M. Jamei, "An intelligent and energy efficient area coverage protocol for wireless sensor networks," *International Journal of Grid and Distributed Computing*, 2011.

[14] H. R. Mohammad Amin Zare Soltani, Abolfazl Toroghi Haghighat and T. G. Chegini, "A couple of algorithms for k-coverage problem in visual sensor networks," *International Conference on Communication Engineering and Networks*, 2011.

[15] M. Cardei, M. T. Thai, Y. Li, and W. Wu, "Energy-efficient target coverage in wireless sensor networks," in *in IEEE Infocom*.

[16] X. Li, H. Frey, N. Santoro, and I. Stojmenovic, "Localized sensor self-deployment with coverage guarantee," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 12, pp. 50–52, Apr. 2008.

[17] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization in distributed sensor networks," *ACM TRANSACTIONS ON EMBEDDED COMPUTING SYSTEMS*, vol. 3, no. 1, pp. 61–91, 2004.

[18] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces," in *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications INFOCOM 2003.*

[19] G. Wang, G. Cao, and T. La Porta, "Movement-assisted sensor deployment," in *Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2004.*

[20] M. Ma and Y. Yang, "Adaptive triangular deployment algorithm for unattended mobile sensor networks," *Computers, IEEE Transactions on*, 2007.

[21] S. L. X. Bai and J. Xu, "Mobile sensor deployment optimization for k -coverage in wireless sensor networks with a limited mobility model," *IETE Technical Review*, 2010.

[22] J.-P. Sheu, G.-Y. Chang, and Y.-T. Chen, "A novel approach for k-coverage rate evaluation and re-deployment in wireless sensor networks," in *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, pp. 1–5, 2008.

[23] G. G. Wang, G. Cao, P. Berman, and T. F. L. Porta, "Bidding protocols for deploying mobile sensors," *IEEE Transactions on Mobile Computing*, vol. 6, no. 5, pp. 563–576, 2007.

[24] N. Bartolini, T. Calamoneri, E. Fusco, A. Massini, and S. Silvestri, "Autonomous deployment of self-organizing mobile sensors for a complete coverage," in *Self-Organizing Systems* (K. Hummel and J. Sterbenz, eds.), vol. 5343 of *Lecture Notes in Computer Science*, pp. 194–205, Springer Berlin Heidelberg, 2008.

[25] G. Tan, S. A. Jarvis, and A.-M. Kermarrec, "Connectivity-guaranteed and obstacle-adaptive deployment schemes for mobile sensor networks," *IEEE Transactions on Mobile Computing*, vol. 8, no. 6, pp. 836–848, 2009.

[26] D. Li, W. Liu, and L. Cui, "Easidesign: An improved ant colony algorithm for sensor deployment in real sensor network system," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pp. 1–5, 2010.

[27] H. J. K. Shohreh Ebrahimnezhad and M. E. Moghaddam, "Extending coverage and lifetime of k-coverage wireless sensor networks using improved harmony search," *Sensors & Transducers*, 2011.

[28] C. Ozturk, D. Karaboga, and B. Gorkemli, "Probabilistic dynamic deployment of wireless sensor networks by artificial bee colony algorithm," *Sensors*, vol. 11, no. 6, pp. 6056–6065, 2011.

[29] G. L. Z.W. Geem, J.-H. Kim, "A new heuristic optimization algorithm: Harmony search," *Simulation*, 2001.

[30] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Applied Mathematics and Computation*, 2007.

[31] M. G. Omran and M. Mahdavi, "Global-best harmony search," *Applied Mathematics and Computation*, 2008.

[32] C.-M. Wang and Y.-F. Huang, "Self adaptive harmony search algorithm for optimization," *Expert Systems with Applications*, 2010.

[33] S. D. D. J. P. Chakraborty, G.G. Roy, "An improved harmony search algorithm with differential mutation operator," *Fundamenta Informaticae*, 2009.

[34] J. L. Q.K. Pan, PN Suganthan and M. Tasgetiren, "A local-best harmony search algorithm with dynamic subpopulations," *Engineering Optimization*, 2010.

[35] B. Wang, "Coverage problems in sensor networks: A survey," *ACM Comput. Surv.*, vol. 43, pp. 32:1–32:53, Oct. 2011.

[36] Y.-C. Wang, C.-C. Hu, and Y.-C. Tseng, "Efficient deployment algorithms for ensuring coverage and connectivity of wireless sensor networks," in *First International Conference Proceedings on Wireless Internet, 2005.*, pp. 114–121, 2005.

[37] X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. H. Lai, "Deploying wireless sensors to achieve both coverage and connectivity," in *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '06, (New York, NY, USA), pp. 131–142, ACM, 2006.