Improving Coverage and Connectivity in Mobile Sensor Networks Using Harmony Search

Shaimaa M. Mohamed
Faculty of Computers and Information
Cairo University
Email: s.mosaad@fci-cu.edu.eg

Haitham S. Hamza
Faculty of Computers and Information
Cairo University
Email: hhamza@fci-cu.edu.eg

Imane A. Saroit
Faculty of Computers and Information
Cairo University
Email: i.saroit@fci-cu.edu.eg

***Abstract*—Dynamic deployment aims at enhancing coverage in wireless sensor networks by redistributing sensor nodes after initial random deployment. In this paper, a harmony search based dynamic deployment (HS-DD) technique is proposed that aims at maximizing both network coverage and connectivity. Furthermore, the performance of the proposed algorithm and a number of the HS-variants in dynamic deployment is studied; namely: Harmony Search-Dynamic Deployment (HS-DD), Improved HS-Dynamic Deployment (IHS-DD), Global HS-Dynamic Deployment (GHS-DD), Differential HS-Dynamic Deployment (DHS-DD) and Self adaptive HS-Dynamic Deployment (SaHS-DH). Simulation results show that GHS-DD achieves the best coverage improvement with the minimum moving distance, while SaHS-DD provides better connectivity with reasonable coverage improvement for dense networks.**

I. Introduction

A wireless sensor network (WSN) is a special type of adhoc network consisting of a large number of small lightweight sensor nodes and one or more sink nodes. WSNs are used in a wide range of applications ranging from structural, environmental and habitat monitoring to military surveillance and intrusion detection. The successful operation of many of these applications depends on an adequate coverage of the target area by the deployed sensor nodes [1].

While deterministic deployment is preferred as it ensures proper coverage of the deployment area, random deployment; however is more practical in some environments, such as [2] [3]. In dense static WSN, scheduling sensor nodes into sleep mode is usually used to ensure full coverage, while extending the lifetime of the network. Recently, the use of mobile sensor nodes either for healing coverage holes or for dynamic deployment has been exploited [4] [5].

Dynamic deployment ensures full coverage by redistributing nodes after initial random deployment. In such cases, node mobility is mainly used to obtain a new stationary configuration with better coverage. Approaches proposed for dynamic deployment can be categorized into virtual forces [6]–[11], computational geometry [12], [13], geometrical patterns [14], and evolutionary computation algorithms; such as particle swarm (PSO) [15]–[17] and artificial bee colony (ABC) [18].

Recently, Harmony search (HS) algorithm has been applied to a number of optimization problems, such as localization [19], path planning [20]–[22], and structural design [23]–[25].

In this paper, the use of harmony search algorithm for enhancing both coverage and connectivity is exploited. Our objective function attempts to optimize both coverage and connectivity. In addition, to the best of our knowledge, this is the first time to consider applying GHS, DHS, and SaHS algorithms in the context of dynamic deployment problem in WSN. Simulation results show that using GHS-DD achieves 89% coverage in deployments which theoretically should have 100% coverage for deployments with $r_c \geq \sqrt{3}r_s$ and 99% coverage for deployments with $r_c = r_s$. . SaHS-DD on the other hand; achieves better connectivity improvement with reasonable coverage improvement for deployments with large number of sensors. Both of them; GHS-DD and SaHS-DD; have a small execution time.

The remainder of the paper is organized as follows. Related work is presented is Section 2. The harmony search and a number of its variants for discrete variables is formally defined in Section 3. The proposed Harmony Search-Dynamic Deployment (HS-DD) algorithm, its variants and the system model are presented in Section 4. Simulation results are presented in Section 5. Finally, concluding remarks are presented in Section 6.

II. Related Work

Coverage in WSN has been studied extensively in the literature [2]–[4], [26], [27]. Recently, mobility have been exploited in Mobile Wireless Sensor Network (MWSN) to maximize coverage among other factors such as connectivity and lifetime. The use of a number of optimization techniques have been exploited. In [2], SRAHS; a sensing radius adjusting protocol using HS; is proposed to enhance coverage in static WSN. The work in [3] studies the performance of HS against PSO to achieve K-coverage in visual sensor networks. HS is found to be faster than PSO with a significant convergence rate. The work in [27] investigates the use of HS and learning automata in adjusting sensing radius. The main objective, however, is topology control and not coverage.

In [4], a deployment algorithm based on IHS for hybrid WSN is proposed. The algorithm considers both k-coverage and connectivity. A greedy algorithm is used afterwards to extend the network lifetime by relocating mobile nodes according to their remaining energy. In [26], a hybrid of firefly and HS is used in robotics path planning for periodic replacement of damaged sensor nodes. The work in [28] proposes a hybrid Binary Differential Evolution Harmony Search Algorithm for the optimal node placement in cluster-based Industrial Wireless Sensor Networks. The problem discussed in [28] is different from the one presented here as node placement is deterministic while in our work we assume an initial random deployment. Their objective function mainly focuses on the setup cost, reliability, and communication load not coverage.

The contributions of this paper are as follows:

1) Propose the use of HS algorithm in dynamic deployment problem.
2) Investigate performance of a number of HS variants (GHS, DHS and SaHS) in the dynamic deployment problem. To our best knowledge, this has not been investigated in the literature in the context of WSN coverage improvement.

III. Harmony Search

Harmony search (HS) is an evolutionary computation optimization algorithm inspired by the experiences of musicians in Jazz improvisation [29]. HS has three main phases: Parameter Initialization, Improvising, and updating (Figure 1). 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 ensures that the best solutions will be carried over to the new population. Pitch adjustment modifies a solution from the HM by adjusting it to a neighboring value while randomization enables the algorithm of exploring new possible solutions. HS generates only one new harmonic which replaces the worst one in the old population.

A. Problem and Parameter Initialization

During parameter initialization step, the following HS parameters are initialized:

- Harmony memory size (HMS), the population size.
- Number of decision variables (N).
- The harmony memory considering rate ($HMCR$), the probability of choosing randomly one of the solutions stored in HM.
- Pitch adjusting rate (PAR), the probability of adjusting the chosen solution.
- Bandwidth (BW).
- The number of iterations (NI).
- The harmony memory (HM) is populated randomly with solution vectors.

The $HMCR$, PAR and BW parameters affect the search process. For example, low $HMCR$ values may lead to slow convergence since only few harmonics are selected while very large values may lead to poor exploration of the search [30]. Likewise, low PAR values leads to exploring small part of the search space leading to slow convergence. Very large values on the other hand is similar to random selection [30].

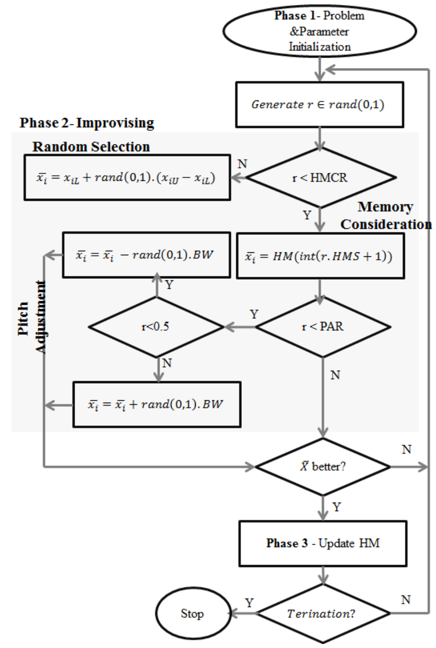


Fig. 1: HS Flowchart

The HM is initialized with HMS vectors each of them represents a possible solution. let X represents a possible solution vector consisting of N decision variables x_i and $f(X)$ is the objective function. Each decision variable can have a lower x_i^L and upper limit x_i^U. Then HM can be initialized as in Eq. 1 [29].

$$HM = \begin{pmatrix} x_1^1 \cdots & x_1^N \mid f(X_1) \\ x_2^1 \cdots & x_2^N \mid f(X_2) \\ \vdots & \vdots \\ x_{HMS}^1 & x_{HMS}^N \mid f(X_{HMS}) \end{pmatrix} \tag{1}$$

where

$$x_i^j = x_i^L + rand(0,1).(x_i^U - x_i^L) \tag{2}$$

and $1 \leq i \leq HMS$,$1 \leq j \leq N$

B. Improvising

During each iteration of the NI iterations, a new solution vector $\overline{X}$ is generated. Each decision variable $\overline{x}_i$ in the solution vector $\overline{X}$ is generated using one of three operations: random selection, memory consideration or pitch adjustment as in Eq. 3.

$$\overline{x}_i = \begin{cases} x_i^L + rand(0,1).(x_i^U - x_i^L) \\ \quad if r < (1 - HMCR) \\ x_i^j \in HM(i) \\ \quad if r < (HMCR).(1 - PAR) \\ x_i^j \pm BW.rand(0,1), x_i^j \in HM(i) \\ \quad if r < (HMCR.PAR) \end{cases} \tag{3}$$

C. Updating the Harmony Memory

If the new harmony vector $\overline{X}$ has a better objective function than any of the harmonics stored in HM. The new harmony replaces the worst harmony vector in HM. A further improvement can be obtained using accidentaling which performs pitch adjustment on each decision variable $\overline{x}_i$ in the new solution vector $\overline{X}$ [31]. The HS algorithm terminates when the maximum number of iterations (NI) is reached.

A number of HS variants have been proposed over the past years [32]–[35]. They either try to improve the algorithm parameter setting step or hybridize HS with other meta-heuristic algorithms [36], [37]. In this paper, the main focus is on parameter setting improvements. HS has three important parameters $HMCR$, PAR and BW. In [32], improved harmony search (IHS) algorithm changes PAR and BW values dynamically each iteration using equation 4, 5.

$$PAR(i) = PAR_{min} + (PAR_{max} - PAR_{min})\frac{i}{NI} \quad (4)$$

$$BW(i) = BW_{max}\exp(\ln(BW_{min}) - \ln(BW_{max}))\frac{i}{NI} \quad (5)$$

Although decreasing BW with an iteration number may improve the final solution, continuous increase in PAR value may lead to oscillation [34]. Another problem is determining the values of the new parameters. Global-best harmony search (GHS) [33]; on the other hand; eliminates the need to use BW parameter by using the concept of “global best particle” from particle swarm optimization (PSO). Improvising is done using equation 6. Although GHS does not need to tune parameter values, it may suffer from premature convergence [34].

$$\overline{x}_i = \begin{cases} x_i^L + rand(0,1).(x_i^U - x_i^L) \\ \qquad ifr < (1 - HMCR) \\ x_i^j \in HM(i) \\ \qquad ifr < (HMCR).(1 - PAR) \\ x_r^{best}, r \in rand(0,1) \\ \qquad ifr < (HMCR.PAR) \end{cases} \quad (6)$$

Another attempt to eliminate the need for BW parameter is proposed in [35]. Differential HS (DHS) replaces the pitch adjustment process with a mutation one borrowed from differential evolution (DE) algorithm as in equation 7, where F is a scaling factor, r1 and r2 are two random values.

$$\overline{x}_i = \begin{cases} x_i^L + rand(0,1).(x_i^U - x_i^L) \\ \qquad ifr < (1 - HMCR) \\ x_i^j \in HM(i) \\ \qquad ifr < (HMCR).(1 - PAR) \\ x_i^j + F(x_{r1} - x_{r2}) \\ \qquad ifr < (HMCR.PAR) \end{cases} \quad (7)$$

Another variant is proposed in [34]; namely, the Self Adaptive HS (SaHS). SaHS attempts to avoid parameter setting and enhance the quality of the final solution by using the adaptive PAR as in equation4 and the minimum and maximum values; instead of BW; in the selected harmonic (8) in pitch adjustment. In addition, it uses low-discrepancy sequences [38] in initializing the HM to reduce the convergence time. Table 1 compares the basic HS and its four variants reviewed in this paper.

$$\overline{x}_i = \begin{cases} x_i^L + rand(0,1).(x_i^U - x_i^L) \\ \qquad ifr < (1 - HMCR) \\ x_i^j \in HM(i) \\ \qquad ifr < (HMCR).(1 - PAR) \\ x_i^j + [Max(HM(i) - x_i^j]rand(0,1) \\ x_i^j - [x_i^j - Min(HM(i)]rand(0,1) \\ \qquad ifr < (HMCR.PAR) \end{cases} \quad (8)$$

TABLE I: HS and its variants

Algorithm	HM Init.	Parameter Setting	Improvising
HS [29]	R	R	Eq.(3)
IHS [32]	PAR, BW adaptive using (4,5)	R	Eq.(3)
GHS [33]	PAR adaptive using (4), BW not used	R	Chooses best variable instead of pitch adjustment (6)
DHS [35]	BW not used	R	DE mutation instead of pitch adjustment(7)
SaHS [34]	PAR adaptive using (4), BW not used	Low-discrepancy sequences	Pitch adjustment using min. and max. values (8)

IV. System Model and Problem Formulation

This paper studies and compares the performance of applying the HS algorithm and a number of its variants to the dynamic deployment problem in WSNs. Five algorithms are implemented and evaluated; namely HS Dynamic Deployment (HS-DD), IHS Dynamic Deployment (IHS-DD), GHS Dynamic Deployment (GHS-DD), DHS Dynamic Deployment (DHS-DD) and SaHS Dynamic Deployment (SaHS-DD). The main objective is to find new positions of the sensor nodes after initial random deployment that maximizes the full coverage and maintains connectivity in the deployment area.

A. Network Model

We consider a WSN in which a set of mobile sensors S are deployed in a 2D rectangular area A. Sensors are homogeneous, using a binary disk sensing model with radius r_s and a communication range r_c. A point is considered to be k-covered if it is within distance less than at least k sensors' sensing radius. Full coverage; thus, can be defined as follows:

$$C = \cup_{s \in S}\frac{\|S\|}{\|A\|} \quad (9)$$

where $\|A\|$ denotes the area of A and $\|S\|$ denotes the area covered by the sensing radius of sensors.

B. Problem Formulation- Algorithm Parameter

HS and a number of its variants are used to solve the dynamic deployment problem where every solution vector represents a possible deployment. It has decision variables equal to the number of sensor (N), where each decision variable represents (x, y) coordinates of a sensor node. Both one-coverage and one-connectivity are checked for each row of HM and the objective function is defined as follows:

$$Max.(F) = \alpha.C + \beta.Conn. \tag{10}$$

Subject to

$$\begin{aligned} x_i \in X, i = 1, 2, 3, ..., N \\ x_i^L \le x_i \le x_i^U \\ y_i^L \le y_i \le y_i^U \end{aligned} \tag{11}$$

where C denotes coverage and is calculated using Eq. 9. *Conn.* denotes connectivity and is calculated using metric in [39]. α and β are scaling factors. (x_i, y_i) are the values of the x_i decision variable. (x_i^L, y_i^L), (x_i^U, y_i^U) represent the minimum and maximum boundaries of the deployment area. The algorithms parameters are listed in Table II

TABLE II: HS and its variants Parameters

Parameter	Value
HMS	5
NI	500
HMCR	0.9
PAR	0.35
BW	0.01
Min PAR	0.01
Max PAR	0.99
Min BW	0.0001
Max BW	0.0005

V. Theoretical Minimum Number of Sensors

In order to compute a precise bound on the performance of the proposed algorithms, the optimal number of sensors in deterministic patterns is used. The triangle lattice pattern has been proven to be optimal to achieve one-coverage and up to six-connectivity when $r_c/r_s \ge \sqrt{3}$ [40]. However, when $r_c/r_s \le \sqrt{3}$, the strip-based pattern outperforms the triangle lattice and can achieve one-coverage and one/two-connectivity [41]. The minimum number of sensor nodes to cover a square with area L^2, and the centers of which form a 1-connected network is calculated as given in [41] as follows:

$$\lim_{r_s \to 0} \pi r_s^2 N(r_s, r_c) = K(r_s, r_c) L^2 \tag{12}$$

Where

$$K(r_s, r_c) = \begin{cases} 2\pi\sqrt{3}/9 \\ \quad if \pi/3 \ge \varphi \\ \pi(\sin\varphi + 2\sin\frac{\pi - 2\varphi}{2})^{-1} \\ \quad if \pi/3 \le \varphi \end{cases}$$

By applying Eq. 12 to our deployment area (100 x 100 m^2) and $r_s = 10m$, the required minimum theoretical number of nodes to achieve one-coverage and up to six-connectivity when $r_c \ge \sqrt{3} r_s$ is 40 and 170 nodes, respectively, when $r_c = r_s$.

VI. Simulation Results

A simulator is implemented in Java to evaluate the performance of the proposed HS-DD algorithm and its variants. In our simulation, sensors are placed randomly and uniformly over a 100 x 100 m^2 area. The sensing radius of each sensor is 10 m. Two cases are considered: (1) equal sensing and communicating radius, and (2) the communication radius is $\sqrt{3}$ of the sensing radius where coverage implies connectivity [39]. For simplicity, network channel is assumed to be error-free and collision-free. In order to compare the performance of the different algorithms, the number of deployed sensors is increased from 30 to 210, and the results are obtained by averaging over 20 simulation runs. The theoretical number of sensors explained in section V is used as a baseline to evaluate the performance of the algorithms under consideration.

The evaluation considers the effect of the scaling factors α and β on both coverage and connectivity.

A. *Effect of Scaling Factors*

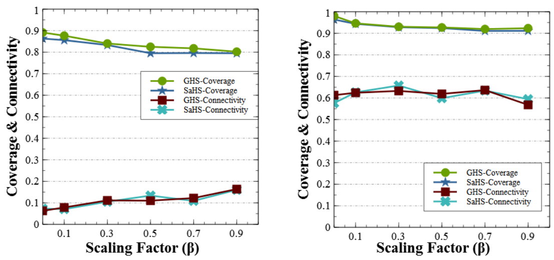
Two metrics are used to evaluate the performance of the algorithms, namely, coverage as defined in Eq. (9), and reachability [39].

In the experiments, the objective is to study the effect of the scaling factors α and β on both coverage and connectivity. In order to do so, the scaling factors, α is set to 1 while β is varying from 0 to 0.9. With $\beta = 0$ means maximizing coverage only, while setting $\beta = 0.9$ attempts to maximize both coverage and connectivity. Since in these experiments $r_c \le \sqrt{3} r_s$, large number of sensor nodes is needed to achieve both coverage and connectivity.

For small number of sensors (i.e, n=30 and 60), increasing the value of β from 0 to 0.9 improved the connectivity of the network but at the expense of reducing the achieved coverage as shown in Figure 2-(a). For number of sensors between 90 and 150, where reasonable overlapping between sensor nodes can be achieved, varying β has a small effect on both coverage and connectivity as shown in Figure 2-(b). While for large number of sensors (i.e, n$\ge$180) increasing β improves connectivity without affecting coverage as shown in Figure2-(c). Considering the trade off between coverage and connectivity, a β value of 0.5 gives good results for all number of sensors.

GHS-DD achieves the best coverage improvement among the five algorithms as can be seen from Table III. Considering the trade off between coverage and connectivity; GHS-DD is more suitable for small to moderate number of sensors (n=30-150), where there is a significant improvement in coverage and fair one in connectivity. SaHS-DD, on the other hand, achieves better results for large number of sensors (n $\ge$ 180), where the connectivity improvement is more significant than the improvement achieved in coverage.

In deployments with 180 nodes, which theoretically should have full coverage and connectivity, HS-DD achieves 98%-coverage and 89%-connectivity, IHS-DD, DHS-DD and SaHS-DD achieve 97%-coverage and 78%, 89% and 81%, respec-



(a) n=60 (b) n=120

(c) n=210

Fig. 2: Coverage rate and connectivity for $\alpha = 1, r_c = r_s$.

tively, for connectivity. GHS-DD achieves 99%-Coverage and 77%-connectivity.

TABLE III: Coverage Improvement Percentages (n=30, $r_c = r_s$)

Algorithm	$\beta = 0$	$\beta = 0.5$	$\beta = 0.9$
HS-DD	23.9 %	16.7 %	11.7 %
IHS-DD	23.3 %	18.3 %	12.6 %
GHS-DD	27.9 %	18.9 %	12.7 %
DHS-DD	20.9 %	17.1 %	10.3 %
SaHS-DD	19.2 %	17.1 %	10.4 %

Figure 3 shows the coverage improvement when setting the communication radius greater that $\sqrt{3}$ of the sensing radius. In this case, $\alpha = 1, \beta = 0$ as coverage implies connectivity. As can be seen, GHS-DD still outperforms all other algorithms for n=30-150, while SaHS achieves slightly higher coverage for large number of sensors, as shown in Table IV.

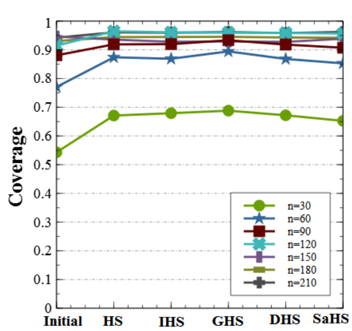


Fig. 3: Coverage rate for $r_c = \sqrt{3} r_s$.

In deployments with 60 sensors, which theoretically should have 100% coverage, HS-DD achieves 87%, both IHS-DD and DHS-DD achieve 86%, SaHS-DD achieves 85% ,while GHS-DD achieves 89%.

TABLE IV: Coverage Improvement Percentages ($r_c = \sqrt{3} r_s$)

Algorithm	$n = 30$	$n = 90$	$n = 210$
HS-DD	23.6 %	4.4 %	1.9 %
IHS-DD	25.1 %	4.4 %	1.8 %
GHS-DD	26.7 %	5.9 %	2.0 %
DHS-DD	23.7 %	4.2 %	1.7 %
SaHS-DD	20.2 %	3.0 %	2.2 %

In terms of execution time, both IHS-DD and SaHS-DD take greater execution time than that of the other algorithms. It is worth noting that both HS-DD and GHS-DD have a reasonable execution time regardless of the number of deployed sensors as shown in Figure 4.

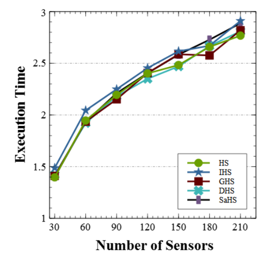
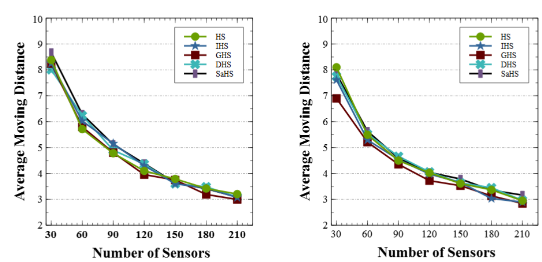


Fig. 4: Execution Time.

Figures 5 and 6 show the average moving distance for different number of sensors and for $r_c = r_s$, $r_c = \sqrt{3} r_s$, respectively. GHS-DD has the minimum moving distance for all different number of sensors for both $r_c = r_s$, and $r_c = \sqrt{3} r_s$.



(a) $\beta = 0.9$ (b) $\beta = 0$

Fig. 5: Average Moving Distance for $r_c = r_s$.

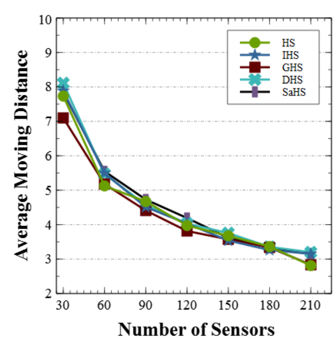


Fig. 6: Average Moving Distance for $r_c = \sqrt{3} r_s$.

VII. Conclusion

In this paper, a dynamic deployment algorithms using harmony search (HS-DD) and a number of its variant are proposed and their performance is evaluated. Simulations show that the GHS-DD outperforms the rest of the algorithms in achieving coverage with minimum moving distance. Our ongoing work investigates how to achieve coverage improvement with the minimum number of moving nodes in order to extend the network's lifetime and study the behavior of the algorithm in achieving k-coverage.

References

[1] 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.

[2] 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.

[3] 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.

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

[5] G. Wang, G. Cao, and T. L. Porta, "Movement-assisted sensor deployment," *Mobile Computing, IEEE Transactions on*, vol. 5, no. 6, pp. 640–652, 2006.

[6] 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.*

[7] 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.*

[8] S. Poduri and G. Sukhatme, "Constrained coverage for mobile sensor networks," in *Proceedings IEEE International Conference on Robotics and Automation, ICRA '04.*

[9] G. Wang, G. Cao, P. Berman, and T. La Porta, "Bidding protocols for deploying mobile sensors," *IEEE Transactions on Mobile Computing,*, 2007.

[10] G. Tan, S. Jarvis, and A.-M. Kermarrec, "Connectivity-guaranteed and obstacle-adaptive deployment schemes for mobile sensor networks," *IEEE Transactions on Mobile Computing,*, 2009.

[11] H. Mahboubi, K. Moezzi, A. Aghdam, K. Sayrafian-Pour, and V. Marbukh, "Self-deployment algorithms for coverage problem in a network of mobile sensors with unidentical sensing ranges," in *IEEE Global Telecommunications Conference (GLOBECOM 2010).*

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

[13] N. Bartolini, T. Calamoneri, T. La Porta, and S. Silvestri, "Autonomous deployment of heterogeneous mobile sensors," *IEEE Transactions on Mobile Computing,*, 2011.

[14] N. Bartolini, T. Calamoneri, E. G. Fusco, A. Massini, and S. Silvestri, "Autonomous deployment of self-organizing mobile sensors for a complete coverage," in *Proceedings of the 3rd International Workshop on Self-Organizing Systems*.

[15] X. Wang, S. Wang, and J.-J. Ma, "An improved co-evolutionary particle swarm optimization for wireless sensor networks with dynamic deployment," *Sensors*, 2007.

[16] X. Wang, J.-J. Ma, S. Wang, and D.-W. Bi, "Distributed particle swarm optimization and simulated annealing for energy-efficient coverage in wireless sensor networks," *Sensors*, 2007.

[17] 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.

[18] C. Ozturk, D. Karaboga, and B. Gorkemli, "Probabilistic dynamic deployment of wireless sensor networks by artificial bee colony algorithm," *Sensors*, 2011.

[19] S.-L. M. I.-T. S. S.-S. R.-V. Diana Manjarres, JavierDelSer, "On the design of a novel two-objective harmony search approach for distance and connectivity based localization in wireless sensor networks," *Engineering Applications of Artificial Intelligence*, 2013.

[20] P. Tangpattanakul and P. Artrit, "Minimum-time trajectory of robot manipulator using harmony search algorithm," in *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on*.

[21] R. Falcon, X. Li, A. Nayak, and I. Stojmenovic, "A harmony-seeking firefly swarm to the periodic replacement of damaged sensors by a team of mobile robots," in *IEEE International Conference on Communications (ICC)*.

[22] S. Panov and S. Koceski, "2nd mediterranean conference on embedded computing (meco)," in *Harmony Search Based Algorithm for Mobile Robot Global Path Planning*.

[23] M. Saka and O. Hasanebi, "Adaptive harmony search algorithm for design code optimization of steel structures," in *Harmony Search Algorithms for Structural Design Optimization* (Z. Geem, ed.), vol. 239 of *Studies in Computational Intelligence*, pp. 79–120, Springer Berlin Heidelberg, 2009.

[24] E. F. Hasanebi, O. and M. Saka, "Adaptive harmony search method for structural optimization,," *Journal of Structural Engineering*, 2010.

[25] S. O. Degertekin, "Improved harmony search algorithms for sizing optimization of truss structures," *Computers and Structures*, 2012.

[26] R. Falcon, X. Li, A. Nayak, and I. Stojmenovic, "A harmony-seeking firefly swarm to the periodic replacement of damaged sensors by a team of mobile robots," in *IEEE International Conference on Communications (ICC),*.

[27] S. M. Nikravan, Mohamad; Jameii, "Combining harmony search and learning automata for topology control in wireless sensor networks," *International Journal of Wireless and Mobile Networks*, 2012.

[28] P. G. G. H. W. Ling Wang; Wei Ye, Yunfei Mao and M. Fei, "The node placement of large-scale industrial wireless sensor networks based on binary differential evolution harmony search algorithm," *International Journal of Innovative Computing, Information and Control*, 2013.

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

[30] X.-S. Yang, "Harmony search as a metaheuristic algorithm," *in: Music-Inspired Harmony Search Algorithm: Theory and Applications, Springer*, 2009.

[31] Z. Geem, *Recent Advances in Harmony Search Algorithm*. Springer, 2009.

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

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

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

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

[36] O. Alia and R. Mandava, "The variants of the harmony search algorithm: An overview," *Artificial Intelligence Review*, 2011.

[37] D. A. P. Sachin A. Patil, "An overview: Improved harmony search algorithm and its applications in mechanical engineering," *International Journal of Engineering Science and Innovative Technology (IJESIT)*, 2013.

[38] C. Lcot, "An algorithm for generating low discrepancy sequences on vector computers," *Parallel Computing*, 1989.

[39] I. S. Petur, S., "Reachability: An alternative to connectivity for sparse wireless multi-hop networks," in *In: Poster at IEEE Infocom*.

[40] R. Kershner, "The number of circles covering a set," *American Journal of Mathematics*, 1939.

[41] 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.