KYBERNETIKA — VOLUME *57* (2021), NUMBER 4, PAGES 568–593

SOLVING THE SENSOR COVER ENERGY PROBLEM VIA INTEGER LINEAR PROGRAMMING

Pingke Li

This paper demonstrates that the sensor cover energy problem in wireless communication can be transformed into a linear programming problem with max-plus linear inequality constraints. Consequently, by a well-developed preprocessing procedure, it can be further reformulated as a 0-1 integer linear programming problem and hence tackled by the routine techniques developed in linear and integer optimization. The performance of this two-stage solution approach is evaluated on a set of randomly generated instances and demonstrates that it is capable of solving large size instances of the sensor cover energy problem.

Keywords: sensor coverage problem, max-plus algebra, integer linear programming

Classification: 90C10, 15A80, 52C15

1. INTRODUCTION

An important research topic, generally termed as the coverage problem, in wireless sensor networks is to minimize the number of sensors deployed in a network to cover all target points in a geographical region under some particular coverage requirements. When the energy efficiency or network lifetime is concerned as well, the deployed sensors may be activated alternately with variable sensing ranges whenever possible. Consequently, a set of activated sensors is to be determined along with the sensing radii in order to minimize the energy consumption while preserving a full coverage of target points for monitoring purposes. The sensor coverage problem of different types has been extensively investigated from various aspects, see, e. g., the survey papers [12, 15, 16] and the references therein.

By quantifying the relationship between the energy consumption of a sensor and its sensing radius, the sensor cover energy problem has been formulated by Astorino and Miglionico [1] simply as minimizing the total energy consumption of the sensor network to maintain full coverage under the assumption that the network connectivity is guaranteed by the large enough transmission range. Since the coverage constraints involved in this direct formulation are nonconvex, an algorithm developed for minimizing the difference of two convex functions form is applied by penalizing the constraints into the objective function. However, the algorithm for difference of convex functions

DOI: 10.14736/kyb-2021-4-0568

programming, see, e. g., [7, 8], minimizes a convex approximation of the objective function iteratively and ensures only local optimality in general for the sensor cover energy problem.

Alternatively, as demonstrated by Hoai and Tuy [6], the sensor cover energy problem may be identified after a simple transformation as a special discrete scenario of monotonic optimization and tackled by a generic branch-reduce-and-bound method, originally developed by Tuy et al. [14], in a more efficient manner to search an optimal sensing pattern. This solution approach may solve moderate size instances of the sensor cover energy problem to optimality, by elaborately manipulating the geometrical structure of the feasible domain.

This paper tackles the sensor cover energy problem from an alternative perspective, following the insight of Hoai and Tuy [6]. By exploiting the special structure of the involved constraints, this paper reveals an interesting connection between the sensor cover energy problem and the max-plus linear programming problem. Consequently, a two-stage approach is launched for searching an optimal sensing pattern of the sensor network, which consists of a preprocessing procedure for reformulation and a routine solution method for 0-1 integer linear programming. As demonstrated by the experimental results on randomly generated instances, this two-stage solution method is computationally more efficient and capable of solving large size instances of the sensor cover energy problem.

The rest of this paper proceeds as follows. The sensor cover energy problem is formulated in Sect. 2 and interpreted as a linear programming problem with max-plus linear inequality constraints. It is reformulated into a 0-1 integer linear programming problem in Sect. 3 by a well-developed procedure and ready to be tackled by calling an integer programming solver. The evaluation of this two-stage solution method is reported in Sect. 4 with some concluding comments addressed in Sect. 5.

2. SENSOR COVER ENERGY PROBLEM

Consider m target points to be monitored by a network of n omnidirectional sensors statically deployed in a certain geographical region of interest. Assume that the positions of target points $\boldsymbol{t}_i \in \mathbb{R}^p$, $i = 1, 2, \ldots, m$, and sensors $\boldsymbol{s}_j \in \mathbb{R}^p$, $j = 1, 2, \ldots, n$, are known with p typically being 2 or 3. The energy consumption of a sensor is a monotonically non-decreasing function of its sensing radius, although its exact form depends on the design and the technology in use of the sensor devices. Following the same setting in [1, 6], the energy consumption per time unit of a sensor j, denoted by $E_j(r_j)$, assumes the form

$$E_j(r_j) = \alpha_j r_j^{\beta_j} + \gamma, \qquad \check{r}_j \le r_j \le \hat{r}_j,$$

where the sensing radius r_j is assumed to be in the interval $[\check{r}_j, \hat{r}_j]$ and the positive constants α_j, β_j, and γ are device dependent. The value of β_j ranges typically in the interval $[2, 4]$ as indicated by Zhou et al. [17] and Bartolini et al. [2] and the parameter γ represents the energy consumption of a sensor in the idle state.

Denote, respectively, $M = \{1, 2, \ldots, m\}$ the set of target points to be covered and $N = \{1, 2, \ldots, n\}$ the set of sensors in the network. The sensor cover energy problem concerned in this context is to determine the sensing radius r_j for each sensor $j \in N$

such that each target point $i \in M$ is covered by at least one sensor, i. e., within its sensing range, and the total energy consumption is minimized meanwhile. It hence can be formulated directly as

$$\begin{aligned} \min \quad & E(\boldsymbol{r}) = \sum_{j \in N} E_j(r_j) \\ \text{s.t.} \quad & \\ & \max_{j \in N} \{r_j - \|\boldsymbol{s}_j - \boldsymbol{t}_i\|\} \geq 0, \quad \forall i \in M \\ & \check{r}_j \leq r_j \leq \hat{r}_j, \quad \forall j \in N, \end{aligned}$$

where $\boldsymbol{r} = (r_1, r_2, \ldots, r_n)^T$ is the vector of sensing radii to be determined. The coverage requirement in this formulation is also referred to as 1-coverage, which is routinely adopted when covering a same target point by multiple sensors is not prescriptive. The general k-coverage requirement, which means each target point should be covered by at least k sensors, applies when fault tolerance or stronger monitoring is desired for the sensor network.

Following the same transformation illustrated in [6], denote

$$x_j = \alpha_j r_j^{\beta_j}$$

and

$$a_{ij} = -\alpha_j \|\boldsymbol{s}_j - \boldsymbol{t}_i\|^{\beta_j}$$

for each $i \in M$ and $j \in N$. The 1-coverage requirement on the sensor network may be written equivalently as

$$\max_{j \in N} \{a_{ij} + x_j\} \geq 0, \quad \forall i \in M.$$

Consequently, by omitting the constant term $n\gamma$ in the objective function, the sensor cover energy problem may be formulated as

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) = \sum_{j \in N} x_j \\ \text{s.t.} \quad & \\ & \max_{j \in N} \{a_{ij} + x_j\} \geq 0, \quad \forall i \in M \\ & \ell_j \leq x_j \leq u_j, \quad \forall j \in N, \end{aligned}$$

where $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^T$ is the vector of transformed sensing radii and $\ell_j = \alpha_j \check{r}_j^{\beta_j}$ and $u_j = \alpha_j \hat{r}_j^{\beta_j}$ are the corresponding lower and upper bounds for each $j \in N$. However, from the perspective of max-plus algebra, this is nothing but a linear programming problem with max-plus linear inequality constraints, which implies that such an optimization problem may be routinely handled within the framework of integer and combinatorial optimization and demands no particular solving techniques. Note that this very formulation of the sensor cover energy problem is identified by Hoai and Tuy [6] as a discrete monotonic optimization problem, but not interpreted as a max-plus linear programming problem within the framework of max-plus algebra.

3. INTEGER LINEAR PROGRAMMING REFORMULATION

For the transformed formulation of the sensor cover energy problem developed in Sec. 2, denote, respectively, the data matrix $A = (a_{ij})_{m\times n}$, the zero vector $\mathbf{0} = (0, 0, \ldots, 0)^T$ and unit vector $\mathbf{1} = (1, 1, \ldots, 1)^T$ of compatible sizes, the lower bound vector $\boldsymbol{\ell} = (\ell_1, \ell_2, \ldots, \ell_n)^T$, and the upper bound vector $\boldsymbol{u} = (u_1, u_2, \ldots, u_n)^T$. This optimization problem may be presented in a matrix form as

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) = \mathbf{1}^T\boldsymbol{x} \\ \text{s.t.} \quad & \\ & A \otimes \boldsymbol{x} \geq \mathbf{0} \\ & \boldsymbol{\ell} \leq \boldsymbol{x} \leq \boldsymbol{u}, \end{aligned}$$

where the symbol $\otimes$ stands for the max-plus matrix multiplication in an analogous manner as in conventional linear algebra, i. e., maximization as the addition operation and addition as the multiplication operation. By a translation if necessary, it may be further assumed that $\boldsymbol{\ell} = \mathbf{0}$ and is so assumed for convenience hereinafter in this section to avoid an overload of notations.

The system of max-plus linear inequalities $A\otimes\boldsymbol{x} \geq \mathbf{0}$, along with the bound restriction $\mathbf{0} \leq \boldsymbol{x} \leq \boldsymbol{u}$, has been intensively investigated as well as its other variants in max-plus algebra, see, e. g., the monograph by Butkovič [3]. It is straightforward that for this particular case the solution set, denoted by $S(A; \boldsymbol{u})$, is nonempty if and only if $A\otimes\boldsymbol{u} \geq \mathbf{0}$. Furthermore, it has been known for a long time that the solution set of such a system of max-plus linear inequalities, whenever nonempty, may be characterized as

$$S(A; \boldsymbol{u}) = \bigcup_{\check{\boldsymbol{x}}\in\check{S}(A;\boldsymbol{u})} \{\boldsymbol{x} \mid \check{\boldsymbol{x}} \leq \boldsymbol{x} \leq \boldsymbol{u}\}$$

where $\check{S}(A; \boldsymbol{u})$ is a set consisting of finitely many minimal solutions. Such a structure is also called a copolyblock by Tuy et al. [14] and Hoai and Tuy [6].

Since the objective function $f(\boldsymbol{x}) = \mathbf{1}^T\boldsymbol{x}$ is simply a conventional linear function, it suffices to minimize it over the finite set $\check{S}(A; \boldsymbol{u})$ in order to obtain an optimal solution to the original sensor cover energy problem. Unfortunately, the number of minimal solutions in $\check{S}(A; \boldsymbol{u})$ could be exponentially large and its complete list may require some sophisticated enumeration techniques as illustrated by Fredman and Khachiyan [5] and Elbassioni [4]. Nevertheless, the solution method proposed by Li and Fang [10, 11] and Li [9] can be applied, which, originally developed for fuzzy relational equations, handles the minimal solutions in an implicit manner and results in a 0-1 integer linear programming problem.

For the system of max-plus linear inequalities $A \otimes \boldsymbol{x} \geq \mathbf{0}$ with $\mathbf{0} \leq \boldsymbol{x} \leq \boldsymbol{u}$, its characteristic matrix $\tilde{Q} = (\tilde{q}_{ij})_{m\times n}$, as an essentially equivalent representation, is defined to be an interval-valued matrix as

$$\tilde{q}_{ij} = \begin{cases} [(-a_{ij}) \vee 0, u_j], & \text{if } a_{ij} + u_j \geq 0 \\ \emptyset, & \text{otherwise,} \end{cases}$$

where $\vee$ is the infix notation of maximization. The element $\tilde{q}_{ij}$ of $\tilde{Q}$ contains all the values that the variable x_j may assume to meet the ith inequality without violating the lower and upper bound restrictions. It follows that $S(A;\boldsymbol{u})$ is nonempty if and only if $\tilde{Q}$ contains at least one nonempty element in its each row. Note that all nonempty elements in each column of $\tilde{Q}$ share a common right endpoint and the columns containing only empty elements play no role. Besides, whenever the interval $[0,u_j]$ appears in $\tilde{Q}$, the corresponding row can be deleted to reduce the size of $\tilde{Q}$. Some additional rules for dimension reduction may be found in Li and Fang [10].

Another crucial feature of the characteristic matrix $\tilde{Q}$ is that a minimal solution in $\check{S}(A;\boldsymbol{u})$ assume only the values specified by the lower bound vector $\boldsymbol{\ell}=\mathbf{0}$ and the left endpoints of those nonempty elements in $\tilde{Q}$. Consequently, each variable of the vector $\boldsymbol{x}$ may be coded by an auxiliary set of binary variables when a minimal solution in $\check{S}(A;\boldsymbol{u})$ is desired to minimize the objective function $f(\boldsymbol{x})$. Specifically, denote k_j for each $j\in N$ the number of different positive values in the set

$$\big\{(-a_{ij})\vee 0 \mid a_{ij}+u_j\geq 0,\ i\in M\big\}$$

and list these values, in ascending order by convention, as

$$\check{\boldsymbol{v}}_j=(\check{v}_{j1},\check{v}_{j2},\dots,\check{v}_{jk_j})^T.$$

By this means, using an auxiliary binary vector

$$\boldsymbol{y}_j=(y_{j1},y_{j2},\dots,y_{jk_j})^T\in\{0,1\}^{k_j}$$

for each $j\in N$, the variable x_j is then represented as

$$x_j=\sum_{k\in K_j}\check{v}_{jk}y_{jk}$$

with the cardinality restriction $\sum_{k\in K_j}y_{jk}\leq 1$ where $K_j=\{1,2,\dots,k_j\}$. With a slight abuse of notations, such a representation may be written in a compact form for each $j\in N$ as $x_j=\check{\boldsymbol{v}}_j^T\boldsymbol{y}_j$ and $\mathbf{1}^T\boldsymbol{y}_j\leq 1$ with the dimension of the vector $\mathbf{1}$ being implicitly determined in the context. Subsequently, for each $\check{\boldsymbol{v}}_j$, $j\in N$, an associated binary matrix $Q_j=(q_{ik}^{(j)})_{m\times k_j}$ is constructed as

$$q_{ik}^{(j)}=\begin{cases}1, & \text{if } \check{v}_{jk}\in\tilde{q}_{ij}\\ 0, & \text{otherwise,}\end{cases}$$

by which each column in $\tilde{Q}$ is split according to the left endpoint values of those nonempty elements. According to Li and Fang [10, 11] and Li [9], the system of max-plus linear inequalities $A\otimes\boldsymbol{x}\geq\mathbf{0}$ with $\mathbf{0}\leq\boldsymbol{x}\leq\boldsymbol{u}$, may be reduced to a system of 0-1 integer linear inequalities

$$\begin{aligned}&\sum_{j\in N}Q_j\boldsymbol{y}_j\geq\mathbf{1}\\&\mathbf{1}^T\boldsymbol{y}_j\leq 1,\quad\forall j\in N\\&\boldsymbol{y}_j\in\{0,1\}^{k_j},\quad\forall j\in N\end{aligned}$$

when only the minimal solution set $\check{S}(A;\boldsymbol{u})$ is concerned. It consists of $m+n$ inequalities and a varying number of binary variables, which is determined by the specific pattern of the characteristic matrix $\tilde{Q}$ and up to mn in theory.

As a result, in order to solve the concerned sensor cover energy problem, it suffices to call a well-developed integer programming solver after the introduced preprocessing procedure has been carried out to solve to optimality the following 0-1 integer linear programming problem

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) = \sum_{j\in N} \check{\boldsymbol{v}}_j^T \boldsymbol{y}_j \\ \text{s.t.} \quad & \\ & \sum_{j\in N} Q_j \boldsymbol{y}_j \geq \mathbf{1} \\ & \mathbf{1}^T \boldsymbol{y}_j \leq 1, \quad \forall j \in N \\ & \boldsymbol{y}_j \in \{0,1\}^{k_j}, \quad \forall j \in N \end{aligned}$$

which can be viewed as a set covering problem with some side cardinality constraints and hence is in general an NP-hard problem. An optimal sensing pattern to the original sensor cover energy problem can be constructed accordingly once an optimal solution to the resulted 0-1 integer linear programming problem has been obtained.

4. COMPUTATIONAL EXPERIMENTS

The proposed two-stage solution method for the sensor cover energy problem is evaluated in this section on some test instances randomly generated by the similar rules used in Astorino and Miglionico [1] and Hoai and Tuy [6] for the comparison purpose. Specifically, the target points to be monitored are uniformly distributed over a 100×100 area. The sensors are also randomly distributed over the same area with a maximum sensing radius $\hat{r}_j = 30$ and a minimum sensing radius $\check{r}_j = 0$ for each sensor $j \in N$. The parameters related to the energy consumption are set, respectively, as $\alpha_j = 1$ and $\beta_j = 2$ for each sensor $j \in N$ while the idle-state energy cost γ is not specified which plays no role in determining the optimal sensing pattern.

For the setting of the number m of target points and the number n of sensors, two scenarios are designed with $m = \frac{1}{5}n$ for the nondense case and $m = 2n$ for the dense case.

As an illustration, Figure 1 shows an optimal sensing pattern for 5 target points with 25 sensors provided by the proposed two-stage solution method. It is intuitive that in such a nondense case a target point is very likely to be covered by its nearest sensor, which implies that an instance of such type is somewhat easier to solve.

Besides, Figure 2 illustrates an optimal sensing pattern for 50 target points with 25 sensors, where a target point is usually covered not by its nearest sensor but more or less in a clustering manner together with the target points nearby. Such a clustering feature leads to multiple options when determining the sensing radius for an activated sensor and eventually results in a possibly very large size instance of the 0-1 integer linear programming problem. However, an active sensor may cover, under this particular numerical setting, more than 25% of the whole area if functioning with its maximum

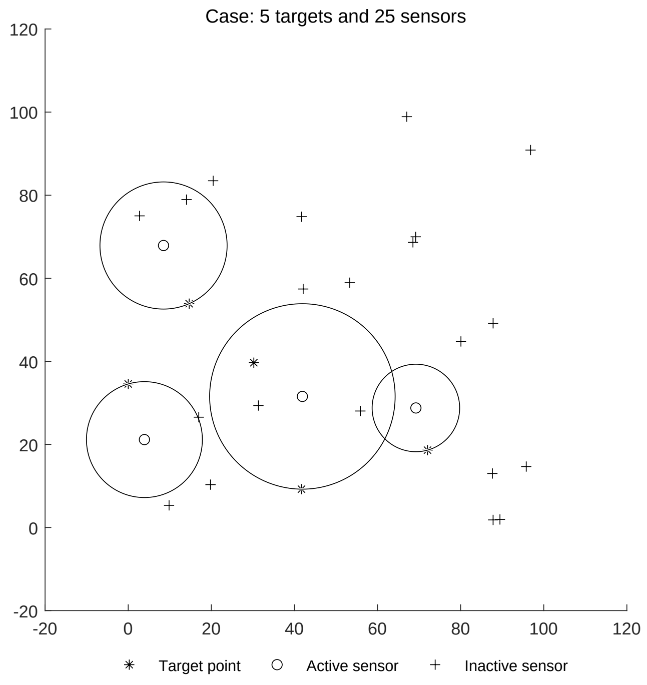

Fig. 1. An optimal sensing pattern for 5 target points with 25 sensors (a nondense case).

sensing radius, which implies that it would cover quite a few target points even in a moderately dense case. Nevertheless, the optimal sensing patten in a dense case still prefers, to some extent, a combination of multiple active sensors each with a small or medium radius to a single one with a large radius, since the energy consumption per time unit is in a quadratic form of the sensing radius. This phenomenon has been witnessed in the numerical examples presented by Astorino and Miglionico [1] and Hoai and Tuy [6] and confirmed as well by the randomly generated instances illustrated in the appendices of this article.

The numerical performance of this two-stage solution method is hence tested on $n = 125$, 250, and 500, for both the nondense and dense cases, each with 5 randomly generated instances. These instances are of much larger sizes compared with those tested in Astorino and Miglionico [1] and Hoai and Tuy [6]. All the instances are run in Matlab R2018b, Win10 64-bit on a laptop with 16G RAM and Intel core i7 2.60GHz. Once an instance has been reformulated into a 0-1 integer linear programming problem, it is tackled by directly calling the solver `intlinprog` from Matlab Optimization Toolbox without additional preprocessing procedure for dimension reduction.

The computational results are documented in Table 1 where the fourth column in-

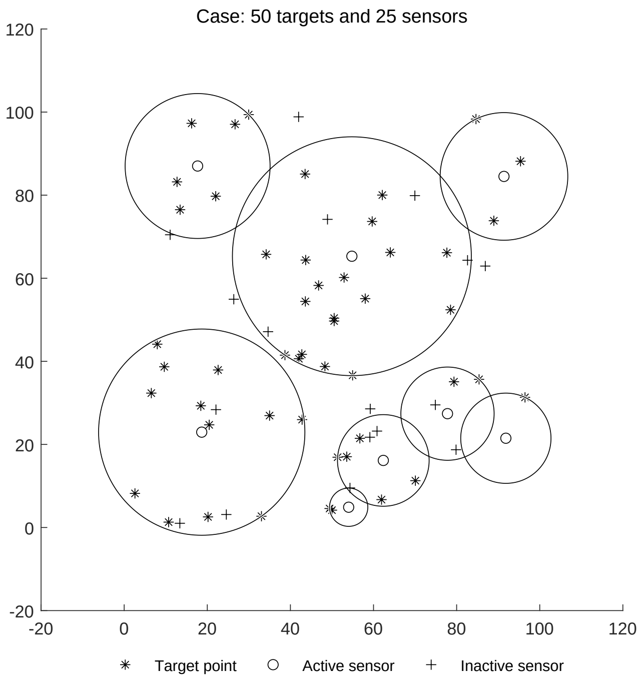

Fig. 2. An optimal sensing pattern for 50 target points with 25 sensors (a dense case).

dexed by 'size' records for each test instance the number of constraints and the number of binary variables in the resulted 0-1 integer linear programming problem. By this two-stage solution method, the computational results demonstrate that the sensor cover energy problem can be solved to optimality in a reasonable time on the dense case of instances with up to 500 sensors and 1000 target points. Moreover, it seems that the resulted 0-1 integer linear programming problem possesses an integer-friendly feature, an emperical phenomenon observed by ReVelle [13] on some types of location-allocation programming problems, by which it means that the corresponding linear programming relaxation often offers a very tight lower bound or even admits an all-integer optimal solution.

Under this particular setting of computational experiments, the proposed two-stage solution method fails to return an optimal solution within a time limit of two hours running the solver `intlinprog` for the dense case of instances with 750 sensors and 1500 target points, which may induce more than 200 thousand binary variables since an active sensor in such a case covers possibly several hundred target points when functioning with its maximum sensing radius. However, an early termination of the solving procedure in the solver `intlinprog` may still provide, thanks to the integer-friendly feature, a near optimal solution according to the lower bound information obtained by solving its linear programming relaxation.

#Sensors	#Targets	Instance	Size	Objective Value[a]	Time[b] (s)	Illustration[c]
125	25	1	150×614	727	0.05	Fig. A1-1
		2	150×615	522	0.05	Fig. A1-2
		3	150×647	696	0.05	Fig. A1-3
		4	150×716	549	0.05	Fig. A1-4
		5	150×619	475	0.05	Fig. A1-5
125	250	1	375×6373	2459	1.6	Fig. B1-1
		2	375×6546	2492	0.9	Fig. B1-2
		3	375×6780	2325	0.9	Fig. B1-3
		4	375×6592	2439	0.9	Fig. B1-4
		5	375×6766	2273	0.9	Fig. B1-5
250	50	1	300×2888	596	0.5	Fig. A2-1
		2	300×2806	505	0.4	Fig. A2-2
		3	300×2633	517	0.4	Fig. A2-3
		4	300×2411	564	0.4	Fig. A2-4
		5	300×2759	649	0.4	Fig. A2-5
250	500	1	750×26844	2454	20.5	Fig. B2-1
		2	750×26112	2381	18.5	Fig. B2-2
		3	750×26860	2415	16.2	Fig. B2-3
		4	750×27186	2480	18.9	Fig. B2-4
		5	750×26535	2368	18.0	Fig. B2-5
500	100	1	600×10868	457	6.1	Fig. A3-1
		2	600×10879	507	6.0	Fig. A3-2
		3	600×11746	607	6.5	Fig. A3-3
		4	600×10976	677	6.2	Fig. A3-4
		5	600×10881	653	6.2	Fig. A3-5
500	1000	1	1500×108464	2231	304.6	Fig. B3-1
		2	1500×108453	2310	510.4	Fig. B3-2
		3	1500×108515	2317	312.6	Fig. B3-3
		4	1500×107787	2283	297.0	Fig. B3-4
		5	1500×107084	2298	307.5	Fig. B3-5

[a] The obtained optimal objective values are rounded up to integers for succinctness.
[b] It includes the time for reformulating, calling the solver `intlinprog` from Matlab Optimization Toolbox, and constructing the sensing radii to the original problem.
[c] The optimal sensing patterns are illustrated in the appendices.

Tab. 1. Computational results of the randomly generated instances.

5. CONCLUSIONS

Taking advantage of the essential discrete features of max-plus linear inequalities, the sensor cover energy problem concerned in this paper is reformulated into a 0-1 integer linear programming problem, which may be viewed as a set covering problem with some side cardinality constraints. Subsequently, demanding no particular solving techniques, it may be solved to optimality, even for large size instances, by directly calling an off-the-shelf integer programming solver. Although such a reformulation approach may substantially increase the problem size as illustrated by the computational experiments on randomly generated instances, the resulted 0-1 integer linear programming problem behaves integer-friendly in the sense of ReVelle [13] that the corresponding linear programming relaxation often provides a very tight lower bound or even admits an all-integer optimal solution. This proposed two-stage solution method is hence capable of handling the sensor cover energy problem in a computationally more efficient manner.

APPENDIX A: ILLUSTRATIONS OF THE NONDENSE INSTANCES

Figure A1-1 to Figure A3-5 illustrate the optimal sensing patterns for the randomly generated nondense instances documented in Table 1, where the number of sensors $n = 125$, 250, and 500, respectively, and the number of target points $m = \frac{1}{5}n$, i.e., $m = 25$, 50, and 100, correspondingly.

APPENDIX B: ILLUSTRATIONS OF THE DENSE INSTANCES

Figure B1-1 to Figure B3-5 illustrate the optimal sensing patterns for the randomly generated dense instances documented in Table 1, where the number of sensors $n = 125$, 250, and 500, respectively, and the number of target points $m = 2n$, i. e., $m = 250$, 500, and 1000, correspondingly.

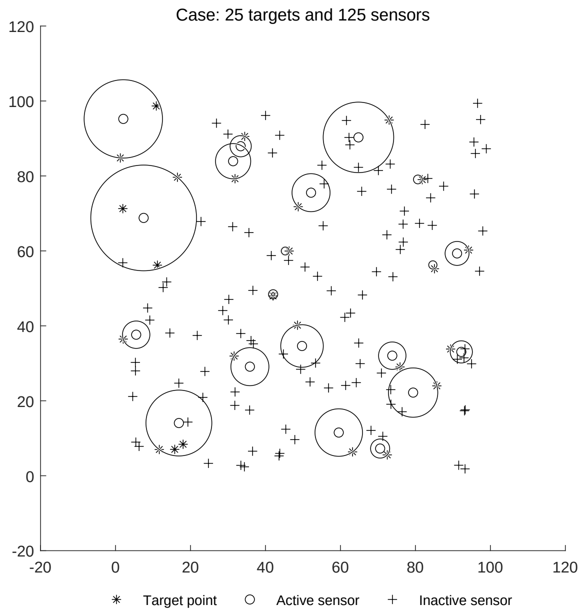

Fig. A1-1. Instance No. 1 for 25 target points with 125 sensors.

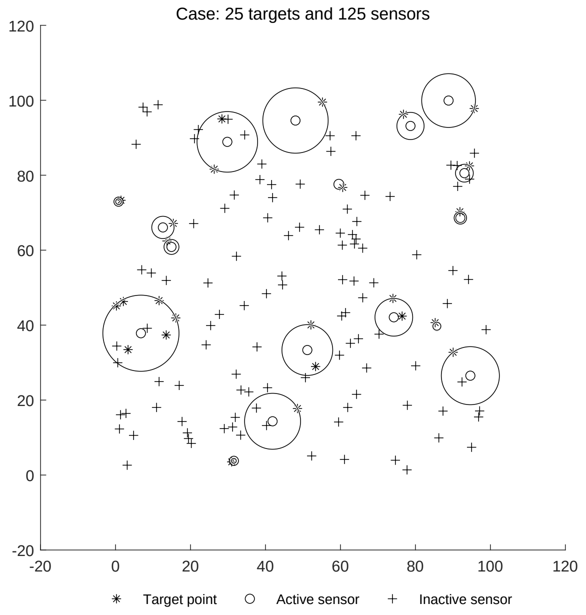

Fig. A1-2. Instance No. 2 for 25 target points with 125 sensors.

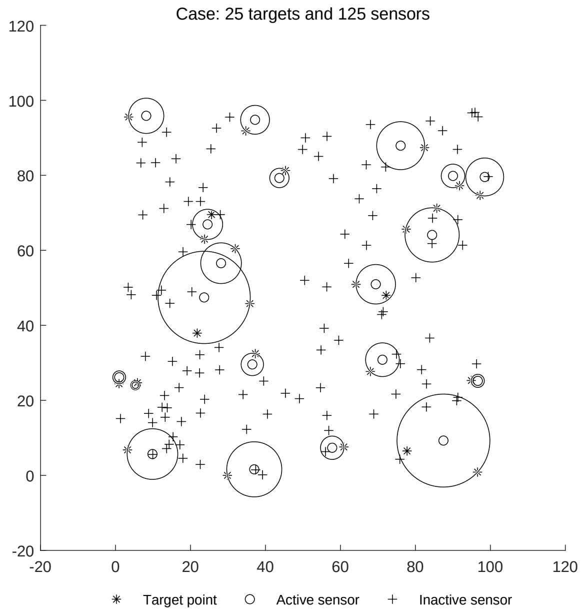

Fig. A1-3. Instance No. 3 for 25 target points with 125 sensors.

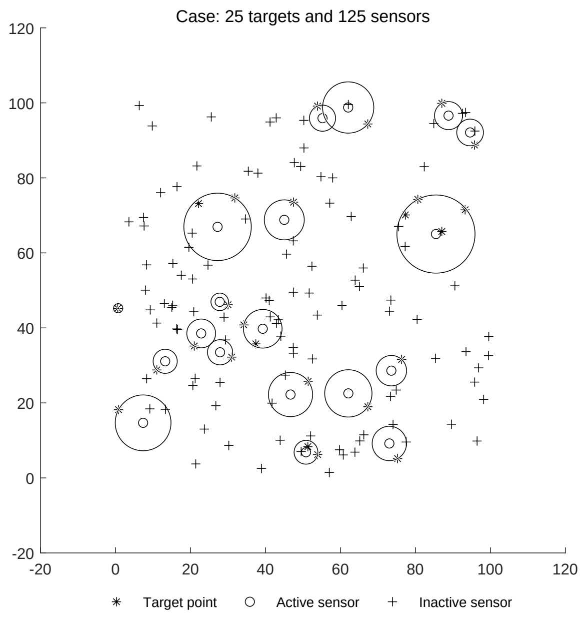

Fig. A1-4. Instance No. 4 for 25 target points with 125 sensors.

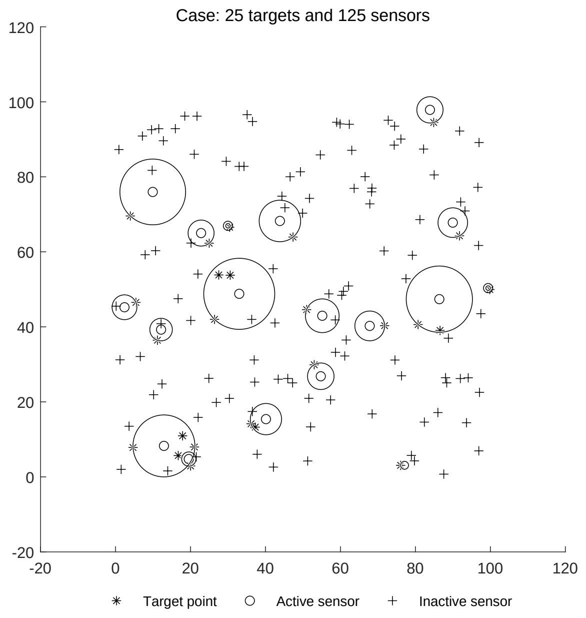

Fig. A1-5. Instance No. 5 for 25 target points with 125 sensors.

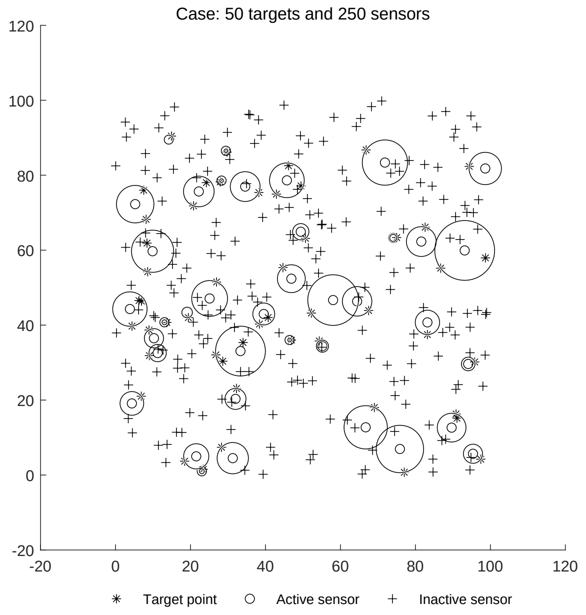

Fig. A2-1. Instance No. 1 for 50 target points with 250 sensors.

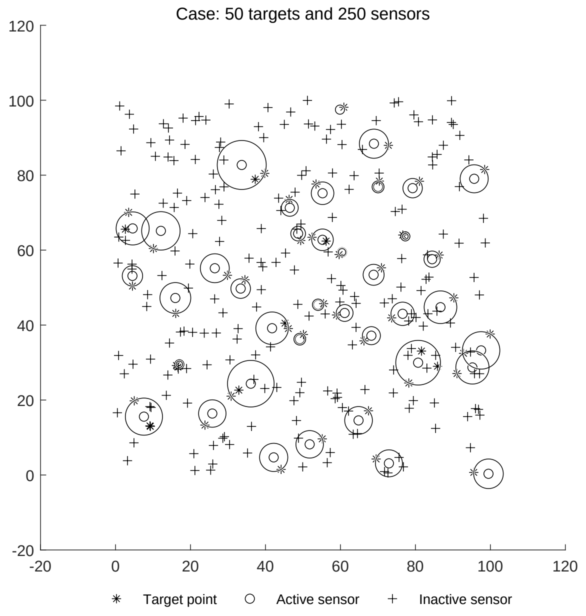

Fig. A2-2. Instance No. 2 for 50 target points with 250 sensors.

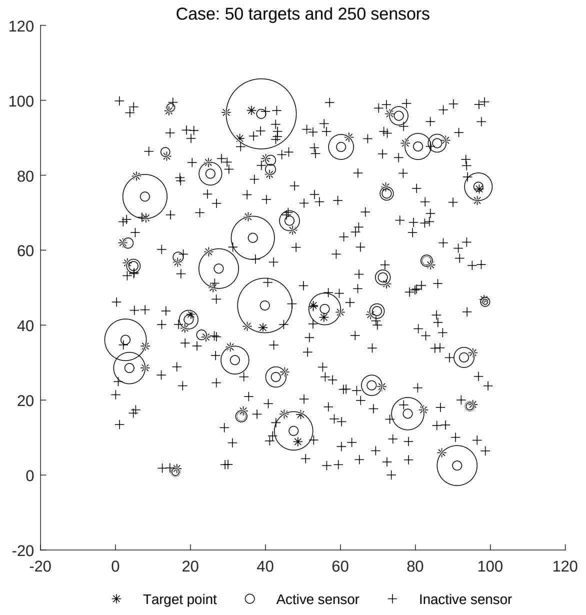

Fig. A2-3. Instance No. 3 for 50 target points with 250 sensors.

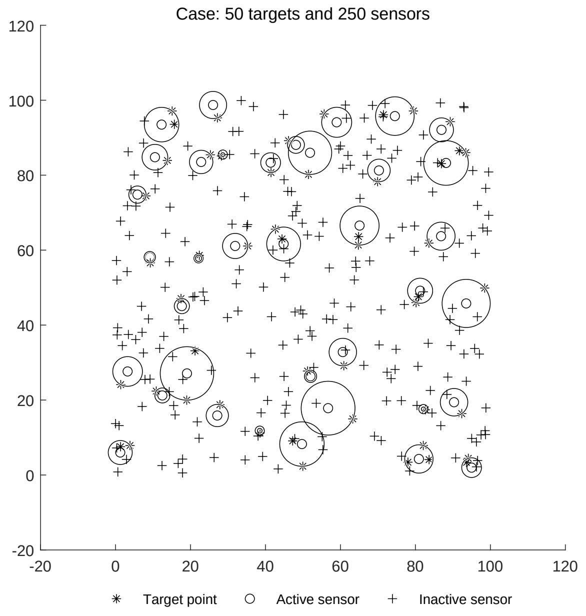

Fig. A2-4. Instance No. 4 for 50 target points with 250 sensors.

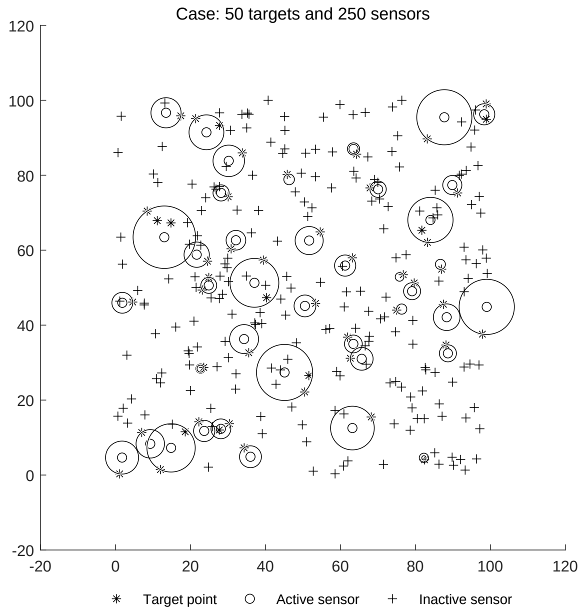

Fig. A2-5. Instance No. 5 for 50 target points with 250 sensors.

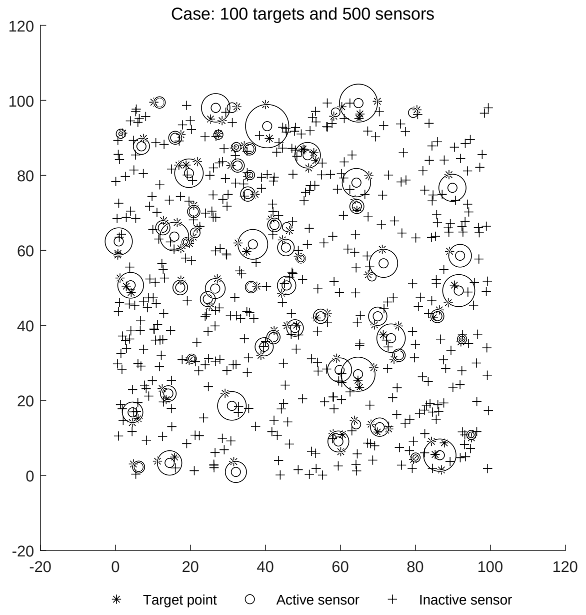

Fig. A3-1. Instance No. 1 for 100 target points with 500 sensors.

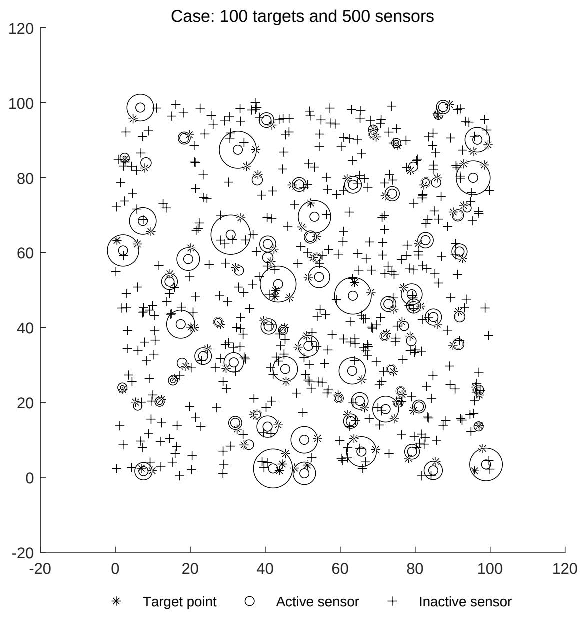

Fig. A3-2. Instance No. 2 for 100 target points with 500 sensors.

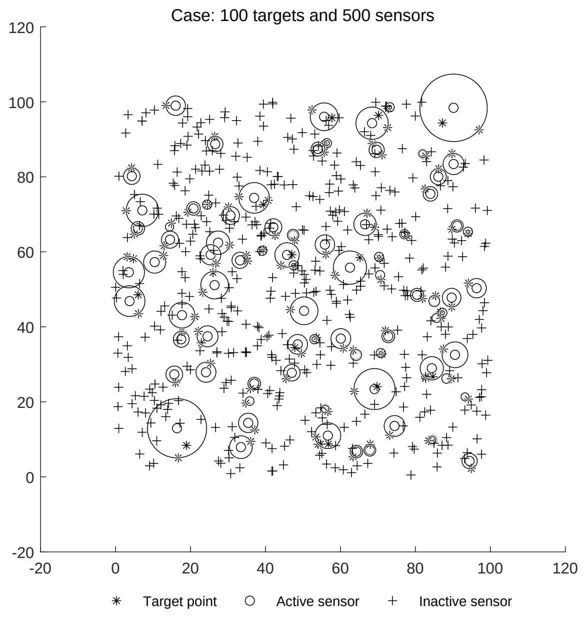

Fig. A3-3. Instance No. 3 for 100 target points with 500 sensors.

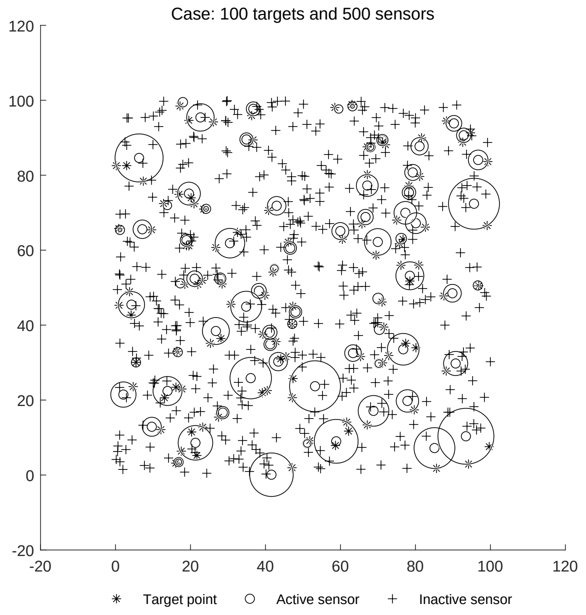

Fig. A3-4. Instance No. 4 for 100 target points with 500 sensors.

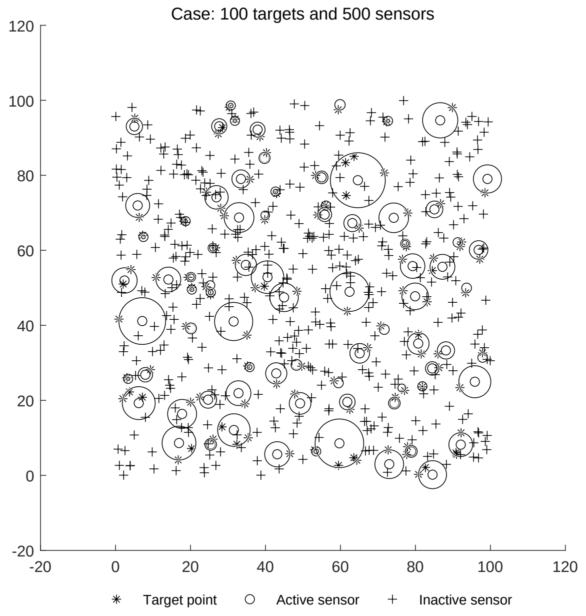

Fig. A3-5. Instance No. 5 for 100 target points with 500 sensors.

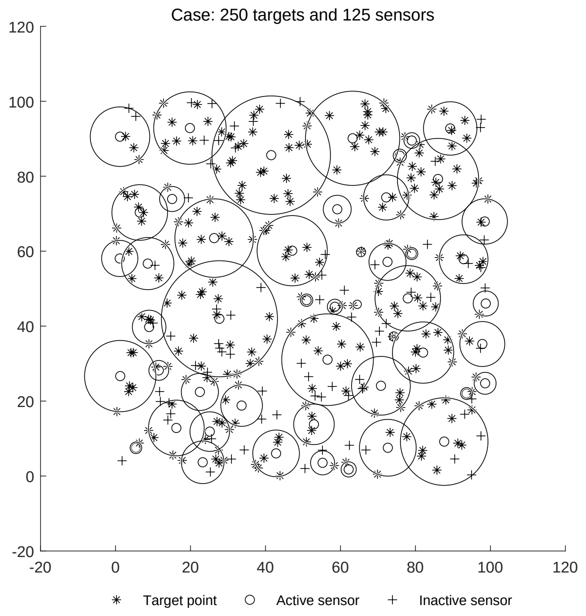

Fig. B1-1. Instance No. 1 for 250 target points with 125 sensors.

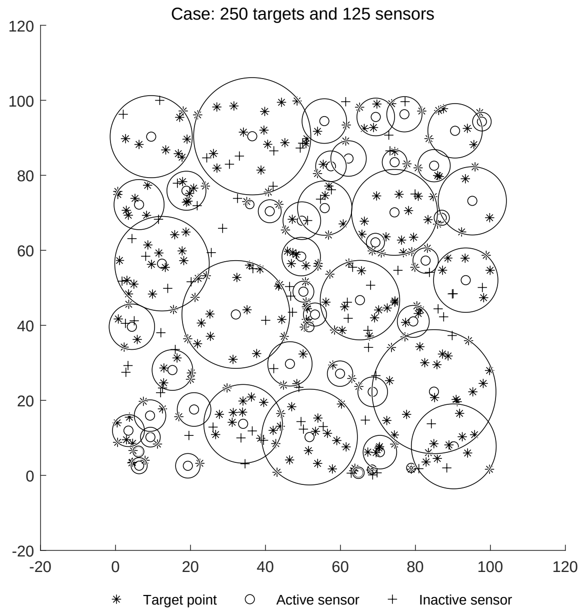

Fig. B1-2. Instance No. 2 for 250 target points with 125 sensors.

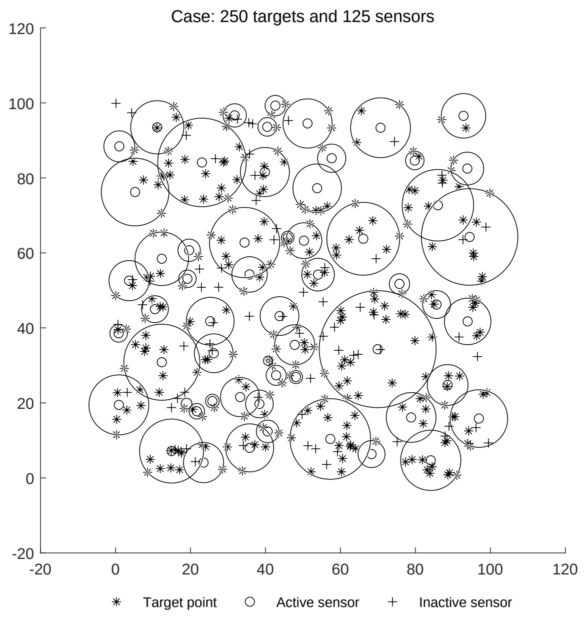

Fig. B1-3. Instance No. 3 for 250 target points with 125 sensors.

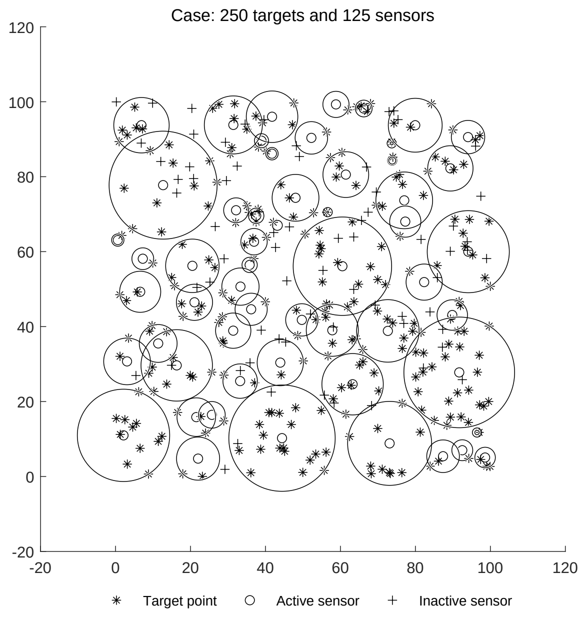

Fig. B1-4. Instance No. 4 for 250 target points with 125 sensors.

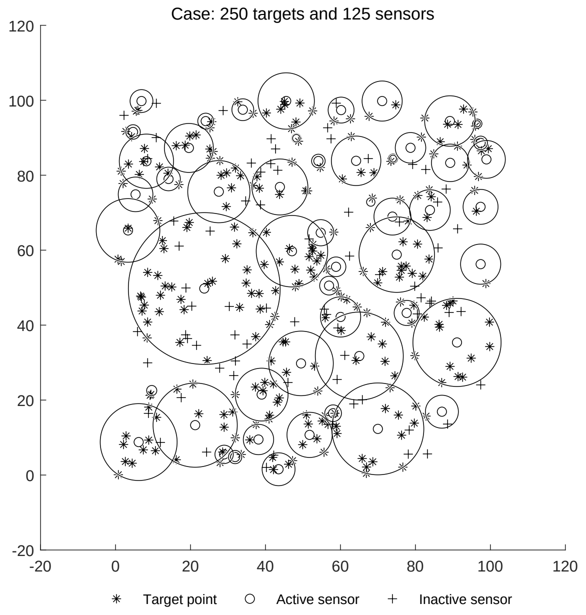

Fig. B1-5. Instance No. 5 for 250 target points with 125 sensors.

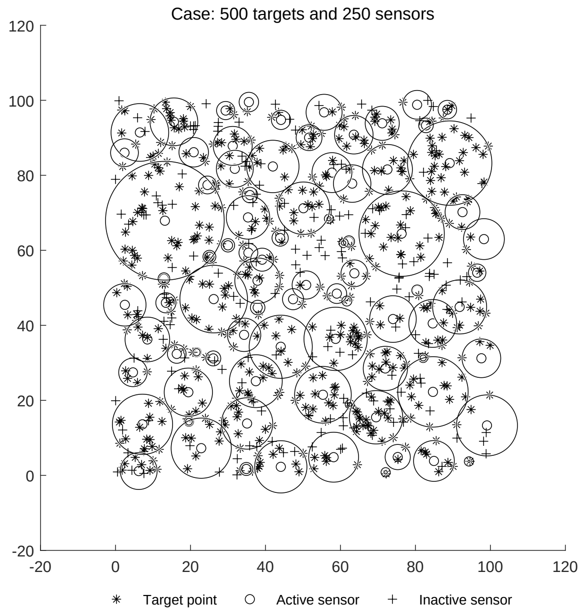

Fig. B2-1. Instance No. 1 for 500 target points with 250 sensors.

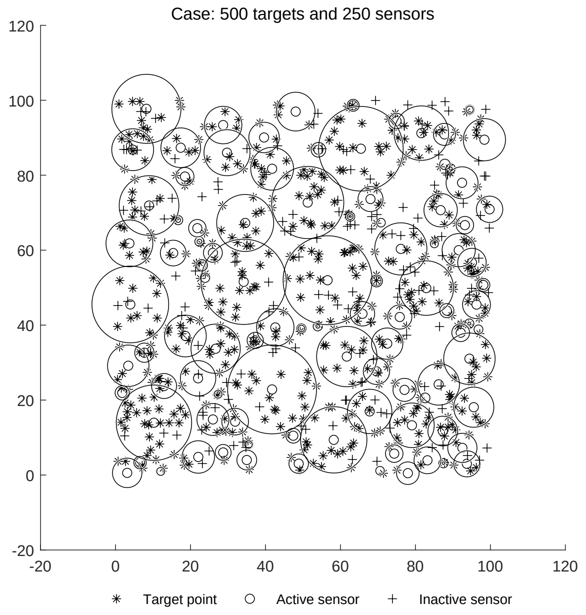

Fig. B2-2. Instance No. 2 for 500 target points with 250 sensors.

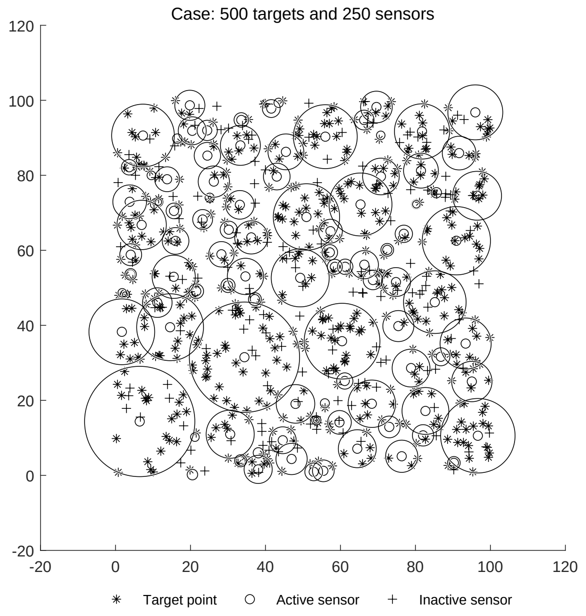

Fig. B2-3. Instance No. 3 for 500 target points with 250 sensors

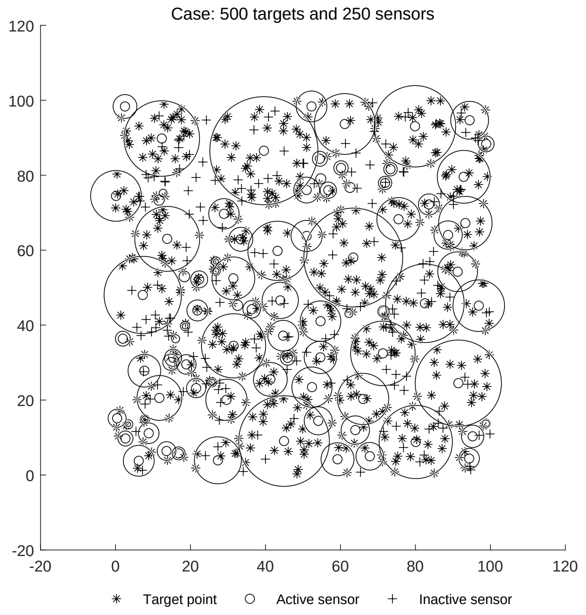

Fig. B2-4. Instance No. 4 for 500 target points with 250 sensors.

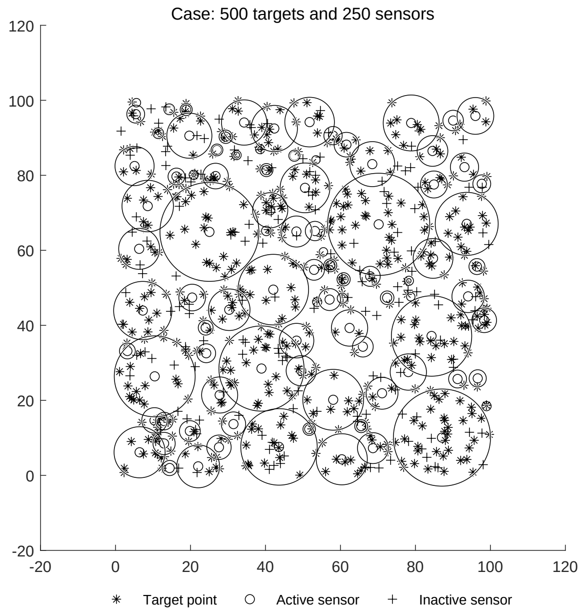

Fig. B2-5. Instance No. 5 for 500 target points with 250 sensors.

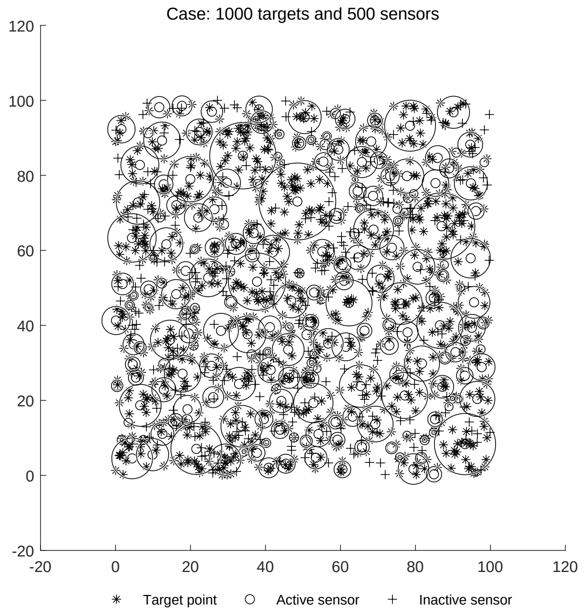

Fig. B3-1. Instance No. 1 for 1000 target points with 500 sensors.

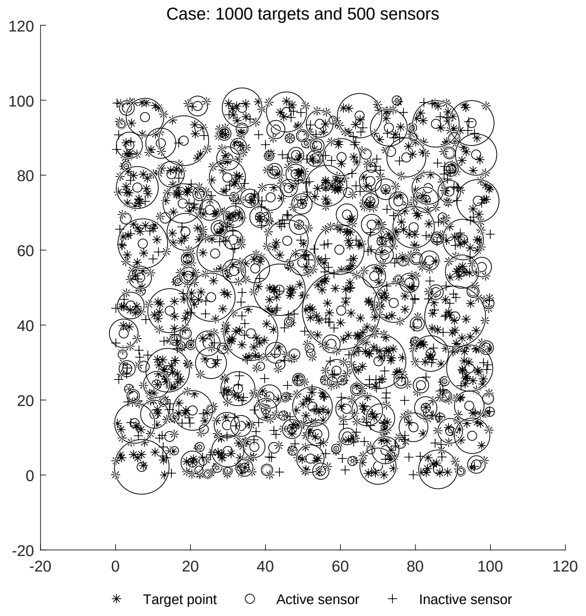

Fig. B3-2. Instance No. 2 for 1000 target points with 500 sensors.

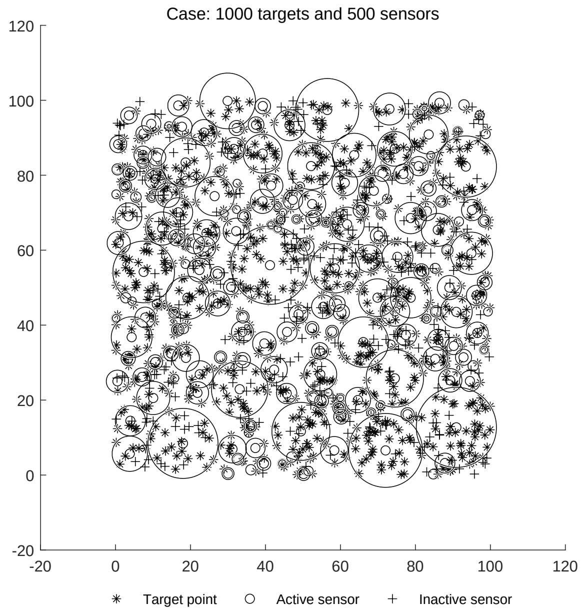

Fig. B3-3. Instance No. 3 for 1000 target points with 500 sensors.

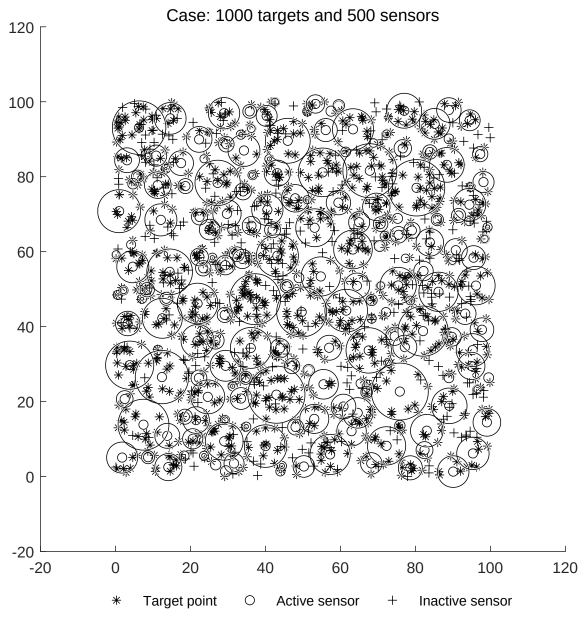

Fig. B3-4. Instance No. 4 for 1000 target points with 500 sensors.

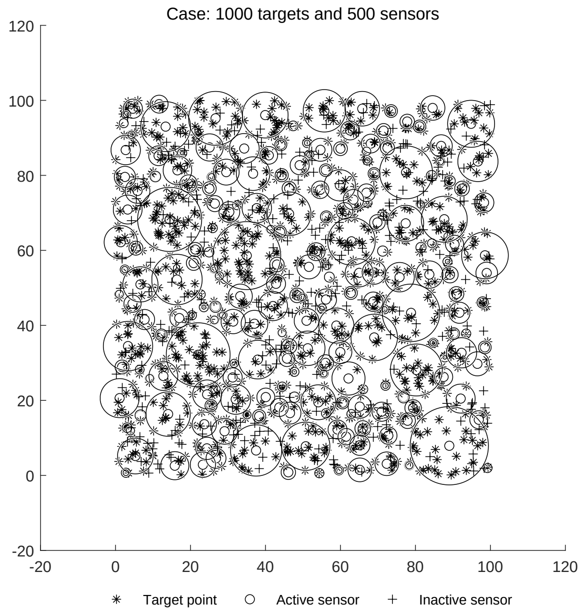

Fig. B3-5. Instance No. 5 for 1000 target points with 500 sensors.

(Received December 7, 2019)

REFERENCES

[1] A. Astorino and G. Miglionico: Optimizing sensor cover energy via DC programming. Optim. Lett. *10* (2016), 2, 355–368. DOI:10.14257/ijsh.2016.10.6.35

[2] N. Bartolini, T. Calamoneri, T. La Porta, C. Petrioli, and S. Silvestri: Sensor activation and radius adaptation (SARA) in heterogeneous sensor networks. ACM Trans. Sensor Netw. *8* (2012), 3, Article 24.

[3] P. Butkovič: Max-linear Systems: Theory and Algorithms. Springer, Berlin 2010.

[4] K. M. Elbassioni: A note on systems with max-min and max-product constraints. Fuzzy Sets Syst. *159* (2008), 2272–2277. DOI:10.1016/j.fss.2008.02.020

[5] M. L. Fredman and L. Khachiyan: On the complexity of dualization of monotone disjunctive normal forms. J. Algorithms *21* (1996), 618–628. DOI:10.1006/jagm.1996.0062

[6] P. T. Hoai and H. Tuy: Monotonic optimization for sensor cover energy problem. Optim. Lett. *12* (2018), 1569–1587. DOI:10.1007/s11590-017-1219-5

[7] H. A. Le Thi and D. T. Pham: The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. Ann. Oper. Res. *133* (2005), 23–46. DOI:10.1007/s10479-004-5022-1

[8] H. A. Le Thi and D. T. Pham: DC programming and DCA: thirty years of developments. Math. Program., Ser. B *169* (2018), 5–68. DOI:10.1007/s10107-018-1235-y

[9] P. Li: Fuzzy Relational Equations: Resolution and Optimization. Ph.D. Dissertation, North Carolina State University 2009. DOI:10.1002/ss.328

[10] P. Li and S.-C. Fang: On the resolution and optimization of a system of fuzzy relational equations with sup-T composition. Fuzzy Optim. Decis. Making *7* (2008), 169–214. DOI:10.1007/s10700-008-9029-y

[11] P. Li and S.-C. Fang: Latticized linear optimization on the unit interval. IEEE Trans. Fuzzy Syst. *16* (2009), 6, 1353–1365.

[12] R. Priyadarshi, B. Gupta, and A. Anurag: Deployment techniques in wireless sensor networks: a survey, classification, challenges, and future research issues. J. Supercomput. *76* (2020), 7333–7373. DOI:10.1007/s11227-020-03166-5

[13] C. S. ReVelle: Facility siting and integer-friendly programming. Eur. J. Oper. Res. *65* (1993), 2, 147–158. DOI:10.1016/0377-2217(93)90329-L

[14] H. Tuy, M. Minoux, and N. T. H. Phuong: Discrete monotonic optimization with application to a discrete location problem. SIAM J. Optim. *17* (2006), 78–97. DOI:10.1137/04060932X

[15] B. Wang: Coverage problems in sensor networks: A survey. ACM Comput. Surv. *43* (2011), 4, Article 32.

[16] Y. Wang, S. Wu, Z. Chen, X. Gao, and G. Chen: Coverage problem with uncertain properties in wireless sensor networks: A survey. Comput. Netw. *123* (2017), 200–232. DOI:10.1016/j.comnet.2017.05.008

[17] Z. Zhou, S.R. Das, and H. Gupta: Variable radii connected sensor cover in sensor networks. ACM Trans. Sensor Netw. *5* (2009), 1, Article 8.

Pingke Li, Department of Industrial Engineering, Tsinghua University, Beijing, 100084. P. R. China.
e-mail: pkli@tsinghua.edu.cn