A Swarm Intelligence Approach for Emergency Infrastructure Inspection Scheduling

Vagelis Plevris, Matthew G. Karlaftis, and Nikos D. Lagaros

Abstract. Natural hazards such as earthquakes, floods and tornadoes can cause extensive failure of critical infrastructures including bridges, water and sewer systems, gas and electricity supply systems, and hospital and communication systems. Following a natural hazard, the condition of structures and critical infrastructures must be assessed and damages have to be identified; inspections are therefore necessary since failure to rapidly inspect and subsequently repair infrastructure elements will delay search and rescue operations and relief efforts. The objective of this work is scheduling structure and infrastructure inspection crews following an earthquake in densely populated metropolitan areas. A model is proposed and a decision support system is designed to aid local authorities in optimally assigning inspectors to critical infrastructures. A combined Particle Swarm – Ant Colony Optimization based framework is developed which proves an instance of a successful application of the philosophy of bounded rationality and decentralized decision-making for solving global optimization problems.

1 Introduction

Infrastructure networks are vital for the well-being of modern societies; national and local economies depend on efficient and reliable networks that provide added value and competitive advantage to an area's social and economic growth. The significance

Vagelis Plevris · Nikos D. Lagaros
Institute of Structural Analysis & Seismic Research,
National Technical University of Athens, 9, Iroon Polytechniou Str.,
Zografou Campus, GR-15780 Athens, Greece
e-mail: {vplevris,nlagaros}@central.ntua.gr

Matthew G. Karlaftis
Department of Transportation Planning and Engineering,
National Technical University of Athens, 9, Iroon Polytechniou Str.,
Zografou Campus, GR-15780 Athens, Greece
e-mail: mgk@central.ntua.gr

K. Gopalakrishnan & S. Peeta (Eds.): Sustainable & Resilient Critical Infrastructure Sys., pp. 201–230.
springerlink.com

of infrastructure networks increases when natural disasters occur since restoration of community functions is highly dependent on the affected regions receiving adequate relief resources. Infrastructure networks are frequently characterized as the most important lifelines in cases of natural disasters; recent experience from around the World (hurricanes Katrina and Wilma, Southeastern Asia Tsunami, Loma Prieta and Northridge earthquakes and others) suggests that, following a natural disaster, infrastructure networks are expected to support relief operations, population evacuation, supply chains and the restoration of community activities.

Infrastructure elements such as bridges, pavements, tunnels, water and sewage systems, and highway slopes are highly prone to damages caused by natural hazards, a result of possible poor construction or maintenance, of design inconsistencies or of the shear magnitude of the natural phenomena themselves. Rapid network degradation following these disasters can severely impact both short and long run operations resulting in increased fatalities, difficulties in population evacuation and the supply of clean water and food to the affected areas. Much of the state-of-the-art in this research area indicates that attention must be given to three important actions: (i) Failsafe design and construction of infrastructure facilities; (ii) Effective maintenance and management of the available facilities; and, (iii) Planning and preparing actions to deal with rapid reparation of infrastructure following the disasters.

As can be expected, significant research has been undertaken in emergency response to either natural hazards or manmade disasters. Work has concentrated on the four main aspects of the process; *mitigation*, *preparedness*, *response*, and *recovery* (an excellent collection of emergency response papers, with a heavy focus on quantitative approaches and algorithms, can be found in Altay and Green [1]). Work on mitigation includes assessing seismic hazards [2], probabilistic damage projection [3-4], and simulation based DSS for integrating the emergency process [5-6]. Research on preparedness, a particularly challenging area of network related problems, has mainly focused on preparing infrastructure networks for dealing with potential disasters and for accommodating evacuation needs [7-12]. Response related work has evolved around two main research paths; first, planning the response-relief logistics operations [13-16], and, second, assessing the performance of the infrastructure system following the natural hazard [17-20]. Finally, recovery operations have attracted limited attention despite their importance in practice; for example, work has concentrated on infrastructure element protection [21], general assessment of relief performance [22], and fund allocation for infrastructure repairs following disasters [23].

It is interesting to note that most research on emergency response, particularly following the disaster, has shied away from dealing with the critical step of damage assessment and its related issues. For example, following an earthquake, all infrastructure elements need to be inspected, damages assessed, and repairs prioritized; these needs pose sets of problems such as partitioning the damaged area into sub-areas of responsibility for repair crews, determining inspection sequences (i.e. which infrastructure elements should be inspected first, second, and so on), and allocating funds for repairs, that research has largely ignored to date. This chapter is focused on issues that are related to inspecting and repairing

infrastructure elements damaged by earthquakes, a highly unpredictable natural disaster of considerable importance to many areas around the World. An explicit effort is made to initiate the development of a process for handling post-earthquake emergency response in terms of optimal infrastructure condition assessment, based on a combined Particle Swarm Optimization (PSO) – Ant Colony Optimization (ACO) framework. Some of the expected benefits from this work include improvements in infrastructure network restoration times and minimization of adverse impacts from natural hazards on infrastructure networks.

2 Notation and Symbols

2.1 Optimization (Generally)

$\boldsymbol{x}$	Design variables vector
$f(\boldsymbol{x})$: $\mathrm{R}^n \rightarrow \mathrm{R}$	Objective function
$\boldsymbol{g}(\boldsymbol{x})$: $\mathrm{R}^n \rightarrow \mathrm{R}^m$	Vector of m inequality constraint functions
$\boldsymbol{x}^{\mathrm{L}}, \boldsymbol{x}^{\mathrm{U}}$	Vectors of length n defining the lower and upper bounds of the design variables, respectively

2.2 Optimum Assignment Problem Definition

$n_{SB}^{(i)}$	Number of structural blocks allocated to the i^{th} inspection crew
SB_k	k^{th} structural block
C_i	Centre of the i^{th} group of structural blocks (with coordinates x_{Ci} and y_{Ci})
$d(SB_k, C_i)$	Distance between the SB_k building block from the centre of the i^{th} group
$D(k)$	Demand for the k^{th} building block
$\mathcal{G}=(\mathcal{N},\mathcal{A})$	Weighted graph where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of arcs (edges or connections) that fully connects the components of $\mathcal{N}$.
$d_{i,j}$ $(i \neq j)$	The distance between two nodes
$\boldsymbol{p}=\{p(1), \ldots, p(\mathrm{N})\}$	A permutation, a possible solution to the Travelling Salesman Problem (TSP), where $\{p(1), \ldots, p(\mathrm{N})\}$ are the node indices
$L(\boldsymbol{p})$	Total length of a solution to the TSP

2.3 Particle Swarm Optimization

$\boldsymbol{v}^j(t)$	Velocity vector of particle j at time t
$\boldsymbol{x}^j(t)$	Position vector of particle j at time t
$\boldsymbol{x}^{\mathrm{Pb},j}$	Personal 'best ever' position of the j^{th} particle
$\boldsymbol{x}^{\mathrm{Gb}}$	Global best location found by the entire swarm
c_1, c_2	Acceleration coefficients: c_1 - cognitive parameter, c_2 - social parameter

$\boldsymbol{r}_1$, $\boldsymbol{r}_2$	Two random vectors uniformly distributed in the interval [0, 1]
$\circ$	Hadamard product, i.e. element-wise vector or matrix multiplication
w	Inertia weight
w_{max}, w_{min}	Maximum and minimum values of the inertia weight, respectively
$\boldsymbol{v}^{\max}$	Vector containing the maximum allowable absolute velocity for each dimension
NP	Number of particles
n	Dimension of particles
$t_{\max}$	Maximum number of iterations for the termination criterion
k_f	Number of iterations for which the relative improvement of the objective function satisfies the convergence check
f_m	Minimum relative improvement of the value of the objective function
Gbest_t	Best value of the objective function found by the PSO at iteration t

2.4 Ant Colony Optimization

m	Number of ants
$\mathcal{M}^k$	Memory of an ant k currently at node i, contains the nodes already visited
$\mathcal{N}_i^k$	the feasible neighbourhood that is the set of nodes that have not yet been visited by ant k
$p_{i,j}^k$	the probability with which ant k, currently at node i, chooses to go to node j
$\tau_{i,j}$	the amount of pheromone on connection between i and j nodes
a, b	superscript parameters a is parameter to control the influence of $\tau_{i,j}$, β is a parameter to control the influence of $\eta_{i,j}$
$\eta_{i,j}$	a heuristic information that is available a priori, denoting the desirability of connection i,j
$d_{i,j}$	Distance between nodes i and j
ρ	rate of pheromone evaporation
$\mathcal{A}$	the set of arcs (edges or connections) that fully connects the set of nodes
$\Delta\tau^k{}_{i,j}(t)$	the amount of pheromone ant k deposits on the connections it has visited through its tour $\mathcal{T}^k$
$L(\mathsf{T}^k)$	Total length of tour $\mathcal{T}^k$ of ant k

3 Problem Formulation

A general formulation of a nonlinear optimization problem can be stated as follows

$$\begin{aligned} &\min_{x\in R^n} f(\boldsymbol{x}) \quad \boldsymbol{x}=[x_1,\ldots,x_n]^{\mathrm{T}} \\ &\text{Subject to} \\ &g_k(\boldsymbol{x})\le 0 \quad k=1,\ldots,m \\ &\boldsymbol{x}^{\mathrm{L}}\le \boldsymbol{x}\le \boldsymbol{x}^{\mathrm{U}} \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is the design variables vector of length n, $f(\boldsymbol{x})$: $\mathrm{R}^n\rightarrow\mathrm{R}$ is the objective function to be minimized, the vector of m inequality constraint functions $\boldsymbol{g}(\boldsymbol{x})$: $\mathrm{R}^n\rightarrow\mathrm{R}^m$ and $\boldsymbol{x}^{\mathrm{L}}$, $\boldsymbol{x}^{\mathrm{U}}$ are two vectors of length n defining the lower and upper bounds of the design variables, respectively.

The main objective of this work is to formulate the problem of inspecting the structural systems of a city/area as an optimization problem. This objective is achieved in two steps: in the first step, the structural blocks to be inspected are optimally assigned into a number of inspection crews (assignment problem), while in the second step the problem of hierarchy is solved for each group of blocks (inspection prioritization problem). In the formulation of the optimization problems considered in this work, the city/area under investigation is decomposed into N_{SB} structural blocks while N_{IG} inspection crews are considered for inspecting the structural condition of all structural and infrastructure systems of the city/area.

3.1 Step 1: Optimum Assignment Problem

The assignment problem is defined as a nonlinear programming optimization problem as follows

$$\begin{aligned} &\min\sum_{i=1}^{N_{IG}}\sum_{k=1}^{n_{SB}^{(i)}}\left[d(SB_k,C_i)\cdot D(k)\right] \\ &x_{C_i}=\frac{1}{n_{SB}^{(i)}}\sum_{k=1}^{n_{SB}^{(i)}}x_k \\ &y_{C_i}=\frac{1}{n_{SB}^{(i)}}\sum_{k=1}^{n_{SB}^{(i)}}y_k \\ &D(k)=A(k)\cdot BP(k) \end{aligned} \tag{2}$$

where $n_{SB}^{(i)}$ is the number of structural blocks allocated to the i^{th} inspection crew, $d(SB_k,C_i)$ is the distance between the SB_k building block from the centre of the i^{th} group of structural blocks (with coordinates x_{Ci} and y_{Ci}), while $D(k)$ is the demand for the k^{th} building block defined as the product of the building block total area

times the built-up percentage (i.e. percentage of the area with a structure). This is defined as a discrete optimization problem since the design variables $\boldsymbol{x}$ are integer numbers denoting the inspection crews to which each built-up block has been assigned and thus the total number of the design variables is equal to the number of structural blocks and the range of the design variables is $[1, N_{IG}]$.

3.2 Step 2: Inspection Prioritization Problem

The definition of this problem is a typical *Travelling Salesman Problem* (TSP) [24] which is a problem in combinatorial optimization studied in operations research and theoretical computer science. In TSP a salesman spends his time visiting N cities (or nodes) cyclically. Given a list of cities and their - pair-wise - distances, the task is to find a Hamiltonian tour of minimal length, i.e. to find a closed tour of minimal length that visits each city once and only once. For an N city asymmetric TSP if all links are present then there are $(N\text{-}1)!$ different tours. TSP problems are also defined as integer optimization problems, similar to all problems that have been proven to be NP-hard [25].

Consider a TSP with N cities (vertices or nodes). The TSP can be represented by a complete weighted graph $G=(\mathcal{N},\mathcal{A})$, with $\mathcal{N}$ the set of nodes and $\mathcal{A}$ the set of arcs (edges or connections) that fully connects the components of $\mathcal{N}$. A cost function is assigned to every connection between two nodes i and j, that is the distance between the two nodes $d_{i,j}$ ($i \neq j$). In the symmetric TSP, it is $d_{i,j}=d_{j,i}$. A solution to the TSP is a permutation $\boldsymbol{p}=\{p(1), \ldots, p(\mathrm{N})\}$ of the node indices $\{1, \ldots, \mathrm{N}\}$, as every node must appear only once in a solution. The optimum solution is the one that minimizes the total length $L(\boldsymbol{p})$ given by

$$L(\boldsymbol{p}) = \sum_{i=1}^{N-1} \left(d_{p(i),p(i+1)} \right) + d_{p(N),p(1)} \tag{3}$$

Thus, the corresponding prioritization problem is defined as follows

$$\min \left[\sum_{k=1}^{n_{SB}^{(i)}-1} d(SB_k, SB_{k+1}) + d(SB_{n_{SB}^{(i)}}, SB_1) \right], i = 1, \ldots, N_{\mathrm{IG}} \tag{4}$$

where $d(SB_k, SB_{k+1})$ is the distance between building block SB_k and $k+1^{\text{th}}$. The main objective is to define the shortest possible route between the structural blocks that have been assigned in *Step 1* to each inspection group.

4 Solving the Optimization Problems

4.1 Particle Swarm Optimization Algorithm

4.1.1 Introduction to Particle Swarm Optimization

Many probabilistic-based search algorithms have been inspired by natural phenomena, such as Evolutionary Programming, Genetic Algorithms, Evolution

Strategies, among others. Recently, a family of optimization methods has been developed based on the simulation of social interactions among members of a specific species looking for food or resources in general. One of these methods is the *Particle Swarm Optimization* (PSO) [26] method that is based on the behavior reflected in flocks of birds, bees and fish that adjust their physical movements to avoid predators and seek for food. The method has been given considerable attention in recent years among the optimization research community.

A swarm of birds or insects or a school of fish searches for food, resources or protection in a very typical manner. If a member of the swarm discovers a desirable path to go, the rest of the swarm will follow quickly. Every member searches for the best in its locality, learns from its own experience as well as from the others typically from the best performer among them. Even human beings show a tendency to behave in this way as they learn from their own experience, their immediate neighbors and the ideal performers in the society. The PSO method mimics the behavior described above. It is a population-based optimization method built on the premise that social sharing of information among the individuals can provide an evolutionary advantage.

PSO has been found to be highly competitive for solving a wide variety of optimization problems [27-33]. It can handle non-linear, non-convex design spaces with discontinuities. Compared to other non-deterministic optimization methods it is considered efficient in terms of number of function evaluations as well as robust since it usually leads to better or the same quality of results. Its easiness of implementation makes it more attractive as it does not require specific domain knowledge information, while being a population-based algorithm, it can be straight forward implemented in parallel computing environments leading to a significant reduction of the total computational cost. PSO has been successfully applied to many fields, such as mathematical function optimization, artificial neural network training and fuzzy system control.

In a PSO formulation, multiple candidate solutions coexist and collaborate simultaneously. Each solution is called a “particle” that has a position and a velocity in the multidimensional design space. A particle “flies” in the problem search space looking for the optimal position. As “time” passes through its quest, a particle adjusts its velocity and position according to its own “experience” as well as the experience of other (neighbouring) particles. Particle's experience is built by tracking and memorizing the best position encountered. As every particle remembers the best position it has visited during its “flight”, the PSO possesses a memory. A PSO system combines local search method (through self experience) with global search method (through neighbouring experience), attempting to balance exploration and exploitation.

4.1.2 Relationship of PSO with Evolutionary Algorithms (EAs)

PSO shares many similarities with evolutionary computation techniques, such as Genetic Algorithms (GA), but the conceptual difference lies in its definition which is given in a social rather than a biological context. The common features of the two optimization approaches include the population concept of the design vectors, initialization with a population of random solutions, a fitness value to evaluate

performance, searching for optima by updating iterations (generations) based on a stochastic process, no requirement for gradient information or user-defined initial estimates and no guaranteed final success. However, unlike GA, PSO has no genetic operators such as crossover and mutation. In PSO, the potential solutions, fly through the problem space by following a velocity update rule. The information sharing mechanism in PSO is significantly different compared to GA. In GA chromosomes share information with each other, so the whole population moves like one group towards an optimal area. In PSO, only Gbest (the global best particle) communicates the information to the others, forming a one-way information sharing mechanism. Compared to Genetic Algorithms, according to the study of Hassan et al. [34], PSO and GA can both obtain high quality solutions, yet the computational effort required by PSO to arrive to such high quality solutions is less than the corresponding effort required by GA. According to Angeline [35], two main distinctions can be made between PSO and an evolutionary algorithm:

i. EAs rely on three mechanisms in their processing: parent representation, selection of individuals and the fine tuning of their parameters. In contrast, PSO only relies on two mechanisms, since PSO does not adopt an explicit selection function. The absence of a selection mechanism in PSO is compensated by the use of leaders to guide the search. However, there is no notion of offspring generation in PSO as with EAs.
ii. The manipulation of the individuals is different in EAs and PSO. PSO uses an operator that sets the velocity of a particle to a particular direction. This can be seen as a directional mutation operator in which the direction is defined by both the particle's personal best and the global best (of the swarm). If the direction of the personal best is similar to the direction of the global best, the angle of potential directions will be small, whereas a larger angle will provide a larger range of exploration. In contrast, EAs use a mutation operator that can set an individual in any direction (although the relative probabilities for each direction may be different). In fact, the limitations exhibited by the directional mutation of PSO has led to the use of mutation operators similar to those adopted in EAs.

4.1.3 Mathematical Formulation of PSO

Each particle maintains two basic characteristics, velocity and position, in the multi-dimensional search space that are updated as follows

$$\boldsymbol{v}^j(t+1) = w\boldsymbol{v}^j(t) + c_1\boldsymbol{r}_1 \circ \left(\boldsymbol{x}^{\mathrm{Pb},j} - \boldsymbol{x}^j(t)\right) + c_2\boldsymbol{r}_2 \circ \left(\boldsymbol{x}^{\mathrm{Gb}} - \boldsymbol{x}^j(t)\right) \quad (5)$$

$$\boldsymbol{x}^j(t+1) = \boldsymbol{x}^j(t) + \boldsymbol{v}^j(t+1) \quad (6)$$

where $\boldsymbol{v}^j(t)$ denotes the velocity vector of particle j at time t, $\boldsymbol{x}^j(t)$ represents the position vector of particle j at time t, vector $\boldsymbol{x}^{\mathrm{Pb},j}$ is the personal 'best ever' position of the j$^{\mathrm{th}}$ particle, and vector $\boldsymbol{x}^{\mathrm{Gb}}$ is the global best location found by the entire swarm. The acceleration coefficients c_1 and c_2 indicate the degree of confidence in the best solution found by the individual particle (c_1 - cognitive parameter) and by

the whole swarm (c_2 - social parameter), respectively, while $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are two random vectors uniformly distributed in the interval [0, 1]. The symbol "$\circ$" of Eq. (5) denotes the Hadamard product, i.e. the element-wise vector or matrix multiplication.

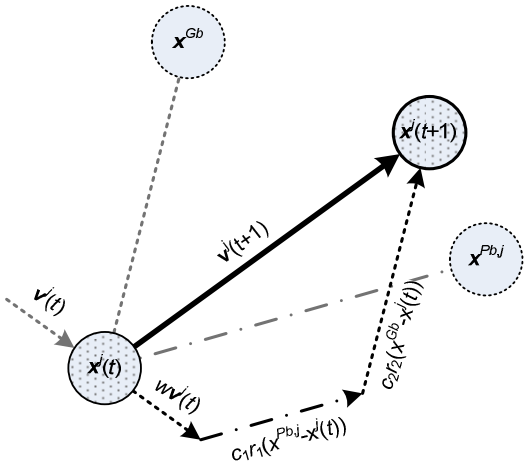


Fig. 1 Visualization of the particle's movement in a two-dimensional design space

Figure 1 depicts a particle's movement, in a two-dimensional design space, according to Eqs. (5) and (6). The particle's current position $\boldsymbol{x}^j(t)$ at time t is represented by the dotted circle at the lower left of the drawing, while the new position $\boldsymbol{x}^j(t+1)$ at time $t+1$ is represented by the dotted bold circle at the upper right hand of the drawing. It can be seen how the particle's movement is affected by: (i) it's velocity $\boldsymbol{v}^j(t)$; (ii) the personal best ever position of the particle, $\boldsymbol{x}^{\mathrm{Pb},j}$, at the right of the figure; and (iii) the global best location found by the entire swarm, $\boldsymbol{x}^{\mathrm{Gb}}$, at the upper left of the figure.

In the above formulation, the global best location found by the entire swarm up to the current iteration ($\boldsymbol{x}^{\mathrm{Gb}}$) is used. This is called a fully connected topology (fully informed PSO), as all particles share information with each other about the best performer of the swarm. Other topologies have also been used in the past where instead of the global best location found by the entire swarm, a local best location of each particle's neighbourhood is used. Thus, information is shared only among members of the same neighbourhood.

The term w of Eq. (5) is the inertia weight, essentially a scaling factor employed to control the exploration abilities of the swarm, which scales the current velocity value affecting the updated velocity vector. The inertia weight was not part of the original PSO algorithm [26], as it was introduced later by Shi and Eberhart [36] in a successful attempt to improve convergence. Large inertia weights will force larger velocity updates allowing the algorithm to explore the design space globally. Similarly, small inertia values will force the velocity updates to concentrate in the nearby regions of the design space.

The inertia weight can also be updated during iterations. A commonly used inertia update rule is the linearly-decreasing, calculated by the formula:

$$w_{t+1} = w_{\max} - \frac{w_{\max} - w_{\min}}{t_{\max}} \cdot t \tag{7}$$

where t is the iteration number, w_{max} and w_{min} are the maximum and minimum values, respectively, of the inertia weight. In general, the linearly decreasing inertia weight has shown better performance than the fixed one.

Particles' velocities in each dimension i (i=1,…,n) are restricted to a maximum velocity $v^{\max}{}_i$. The vector $\boldsymbol{v}^{\max}$ of dimension n holds the maximum absolute velocities for each dimension. It is more appropriate to use a vector rather than a scalar, as in the general case different velocity restrictions can be applied for different dimensions of the particle. If for a given particle j the sum of accelerations of Eq. (5) causes the absolute velocity for dimension i to exceed $v^{\max}{}_i$, then the velocity on that dimension is limited to $\pm v_{max,i}$. The vector parameter $\boldsymbol{v}^{\max}$ is employed to protect the cohesion of the system, in the process of amplification of the positive feedback. The basic PSO has only few parameters to adjust. In Table 1 there is a list of the main parameters, their typical values as well as other information.

Table 1 Main PSO parameters

Symbol	Description	Details
NP	Number of particles	A typical range is 10 – 40. For most problems 10 particles is sufficient enough to get acceptable results. For some difficult or special problems the number can be increased to 50-100.
n	Dimension of particles	It is determined by the problem to be optimized.
w	Inertia weight	Usually is set to a value less than 1, i.e. 0.95. It can also be updated during iterations.
$\boldsymbol{x}^{\mathrm{L}}$, $\boldsymbol{x}^{\mathrm{U}}$	Vectors containing the lower and upper bounds of the n design variables, respectively	They are determined by the problem to be optimized. Different ranges for different dimensions of particles can be applied in general.
$\boldsymbol{v}^{\max}$	Vector containing the maximum allowable velocity for each dimension during one iteration	Usually is set half the length of the allowable interval for the given dimension: $v^{\max}{}_i = (x^{\mathrm{U}}{}_i - x^{\mathrm{L}}{}_i)/2$. Different values for different dimensions of particles can be applied in general.
c_1, c_2	Cognitive and social parameters	Usually c_1=c_2=2. Other values can also be used, provided that $0 < c_1+c_2 < 4$ [26].

4.1.4 Convergence Criteria

Due to the repeated process of the PSO search, convergence criteria have to be applied for the termination of the optimization procedure. Two widely adopted convergence criteria are the maximum number of iterations of the PSO algorithm and the minimum error requirement on the calculation of the optimum value of the objective function. The selection of the maximum number of iterations depends, generally, on the complexity of the optimization problem at hand. The second criterion presumes prior knowledge of the global optimal value, which is feasible for testing or fine-tuning the algorithm in mathematical problems when the optimum is known a priori, but this is certainly not the case in practical structural optimization problems where the optimum is not known a priori.

Table 2 PSO convergence parameters

Symbol	Description	Details
t_{max}	Maximum number of iterations for the termination criterion.	Determined by the complexity of the problem to be optimized, in conjunction with other PSO parameters (n, NP).
k_f	Number of iterations for which the relative improvement of the objective function satisfies the convergence check.	If the relative improvement of the objective function over the last k_f iterations (including the current iteration) is less or equal to f_m, convergence has been achieved.
f_m	Minimum relative improvement of the value of the objective function.	

In our study, together with the maximum number of iterations, we have implemented the convergence criterion connected to the rate of improvement of the value of the objective function for a given number of iterations. If the relative improvement of the objective function over the last k_f iterations (including the current iteration) is less or equal to a threshold value f_m, convergence is supposed to have been achieved. In mathematical terms, denoting as $Gbest_t$ the best value for the objective function found by the PSO at iteration t, the relative improvement of the objective function can be written for the current iteration t as follows

$$\frac{Gbest_{t-k_f+1} - Gbest_t}{Gbest_{t-k_f+1}} \leq f_m \tag{8}$$

In Table 2 there is a list of the convergence parameters of the PSO used in this study with description and details. A pseudo code of the PSO procedure is given in Figure 2.

4.1.5 PSO for Integer Optimization

Since both problems defined in Section 2 are integer optimization problems, discrete optimization algorithms are required. For the Step 1 optimization problem described in Section 2.1, a discrete version of the PSO algorithm is employed. In the continuous version of the PSO method, both particle positions and velocity are initialized randomly.

For each particle j
  Initialize particle position by distributing particles randomly in the design space
End

Repeat

  For each particle j
    Calculate fitness value for current position
    If the current fitness value is better than the best fitness value (*Pbest*) in the particle's history then set current fitness value as the new *Pbest* and current position as the new $\boldsymbol{x}^{\mathrm{Pb}}{}_j$
  End

  Set *Gbest* as the best fitness value of all the particles' *Pbest* and corresponding position as the new $\boldsymbol{x}^{\mathrm{Gb}}$

  For each particle *j*
    Calculate particle velocity from Eq. **(5)**
    Update particle position from Eq. **(6)**
    If, for any dimension i, $x_i \le x^{\mathrm{L}}{}_i$ or $x_i \ge x^{\mathrm{U}}{}_i$, then set $x_i = x^{\mathrm{L}}{}_i$ or $x_i = x^{\mathrm{U}}{}_i$ respectively and set corresponding $v_i = 0$
  End

Until maximum iterations is not attained and the relative improvement of the objective function is greater than f_{m} over the last k_{f} iterations

Report results

Fig. 2 Pseudo-code for the main PSO for unconstrained optimization

In this work, the particle positions are generated randomly over the design space using discrete Latin Hypercube Sampling, thus guaranteeing that the initial particle positions will be integers in the acceptable range. Furthermore, in the case of discrete optimization and in particular in integer programming, at every step of the optimization procedure, integer particle positions should also be generated. In order to satisfy this, Eq. (5) is modified as follows

$$\begin{aligned} &\boldsymbol{v}^j(t+1) = \\ &\text{round}\Big[\, w\boldsymbol{v}^j(t) + c_1\boldsymbol{r}_1 \circ \left(\boldsymbol{x}^{\text{Pb},j} - \boldsymbol{x}^j(t)\right) + c_2\boldsymbol{r}_2 \circ \left(\boldsymbol{x}^{\text{Gb}} - \boldsymbol{x}^j(t)\right)\Big] \end{aligned} \tag{9}$$

where the vector function *round*($\boldsymbol{x}$) rounds each element of the vector $\boldsymbol{x}$ into the nearest integer.

4.2 Ant Colony Optimization

4.2.1 Introduction to Ant Colony Optimization

The *Ant Colony Optimization* (ACO) algorithm [37,38] is a population-based probabilistic technique for solving optimization problems, mainly for finding optimum paths through graphs. The algorithm was inspired by the behaviour of real ants in nature. In many ant species, individuals initially wander randomly and upon finding a food source return to their colony, depositing a substance called *pheromone* on the ground. Other ants smell this substance, and its presence influences the choice of their path, i.e. they tend to follow strong pheromone concentrations rather than travelling completely randomly, returning and reinforcing it if they eventually find food. The pheromone deposited on the ground forms a *pheromone trail*, which allows the ants to find good sources of food that have been previously identified by other ants.

As time passes, the pheromone trails start to evaporate, reducing their strength. The more time it takes for an ant to travel down a path and back again, the more time the pheromone trail has to evaporate. A short path gets marched over faster than a long one, and thus the pheromone density remains high as it is laid on the path faster than it can evaporate. If there was no evaporation, the paths chosen by the first ants would tend to be excessively attractive to the following ants and as a result the exploration of the solution space would be constrained. In that sense, pheromone evaporation helps also to avoid convergence to a locally optimal solution. Positive feedback eventually leads to most of the ants following a single "optimum" path.

The idea of the ant colony algorithm is to mimic this behavior with simulated ants walking around the graph representing the problem to solve. The first algorithm was aiming to search for an optimal path in a graph. The original idea has since diversified to solve a wider class of numerical problems and, as a result, several problems have emerged, drawing on various aspects of the behavior of ants. The initial applications of ACO were in the domain of NP-hard

combinatorial optimization problems, while it was soon also applied to routing in telecommunication networks.

In ACO, a set of software agents called *artificial ants* search for good solutions to the optimization problem of finding the best path on a weighted graph. The ants incrementally build solutions by moving on the graph. The solution construction process is stochastic and it is biased on a *pheromone model*, that is, a set of parameters associated with graph components (either nodes or edges) whose values are modified at runtime by the ants.

The advantages of ACO include its easy implementation, the inherent parallelism of its procedures, the positive feedback that accounts for rapid discovery of good solutions in hard combinatorial optimization problems, its suitability to be used in dynamic applications, e.g. in a TSP where the distances between the nodes change with time, and its great performance with "ill-structured" problems like network routing.

4.2.2 ACO Applied to the TSP

To apply ACO to the TSP, the *construction graph* is considered, defined by associating the set of cities with the set of vertices on the graph. The construction graph is fully connected and the number of vertices is equal to the number of cities, since in the TSP it is possible to move from any given city to any other city. The length of the edges (connections) between the vertices are set to be equal to the corresponding distances between the nodes (cities) and the pheromone values and heuristic values are set for the edges of the graph. Pheromone values are modified during iterations at runtime and represent the cumulated experience of the ant colony, while heuristic values are problem dependent values that, in the case of the TSP, are set to be the inverse of the lengths of the edges.

During an ACO iteration, each ant starts from a randomly chosen vertex of the construction graph. Then, it moves along the edges of the graph keeping a memory of its path. In order to move from one node to another it probabilistically chooses the edge to follow among those that lead to yet unvisited nodes. Once an ant has visited all the nodes of the graph, a solution has been constructed. The probabilistic rule is biased by pheromone values and heuristic information: the higher the pheromone and the heuristic value associated to an edge, the higher the probability the ant will choose that particular edge. Once all the ants have completed their tour, the iteration is complete and pheromone values on the connections are updated: each of the pheromone values is initially decreased by a certain percentage and then it receives an amount of additional pheromone proportional to the quality of the solutions to which it belongs.

4.2.3 Ant Colony Optimization Algorithm

Consider a population of m ants where at each iteration of the algorithm every ant constructs a "route" by visiting every node sequentially. Initially, ants are put on randomly chosen nodes. At each construction step during an iteration, ant k

applies a probabilistic action choice rule, called random proportional rule, to decide which node to visit next. While constructing the route, an ant k currently at node i, maintains a memory $\mathcal{M}^k$ which contains the nodes already visited, in the order they were visited. This memory is used in order to define the feasible neighborhood $\mathcal{N}^k_i$ that is the set of nodes that have not yet been visited by ant k. In particular, the probability with which ant k, currently at node i, chooses to go to node j is

$$p^k_{i,j} = \frac{(\tau_{i,j})^\alpha \cdot (\eta_{i,j})^\beta}{\sum_{\ell \in \mathsf{N}^k_i} \left((\tau_{i,\ell})^\alpha \cdot (\eta_{i,\ell})^\beta \right)}, \quad \text{if } j \in \mathsf{N}^k_i \tag{10}$$

where $\tau_{i,j}$ is the amount of pheromone on connection between i and j nodes, α is a parameter to control the influence of $\tau_{i,j}$, β is a parameter to control the influence of $\eta_{i,j}$ and $\eta_{i,j}$ is a heuristic information that is available a priori, denoting the desirability of connection i,j, given by

$$\eta_{i,j} = \frac{1}{d_{i,j}} \tag{11}$$

According to Eq. (11), the heuristic desirability of going from node i to node j is inversely proportional to the distance between i and j. By definition, the probability of choosing a city outside $\mathcal{N}^k_i$ is zero. By this probabilistic rule, the probability of choosing a particular connection i,j increases with the value of the associated pheromone trail $\tau_{i,j}$ and of the heuristic information value $\eta_{i,j}$.

The selection of the superscript parameters α and β is very important: if α=0, the closest cities are more likely to be selected which corresponds to a classic stochastic greedy algorithm (with multiple starting points since ants are initially randomly distributed over the nodes). If β=0, only pheromone amplification is at work, that is, only pheromone is used without any heuristic bias (this generally leads to rather poor results [38]).

4.2.4 Pheromone Update Rule

After all the m ants have constructed their routes, the amount of pheromone for each connection between i and j nodes, is updated for the next iteration t+1 as follows

$$\tau_{i,j}(t+1) = (1-\rho) \cdot \tau_{i,j}(t) + \sum_{k=1}^{m} \Delta \tau^k_{i,j}(t), \quad \forall (i,j) \in \mathsf{A} \tag{12}$$

where ρ is the rate of pheromone evaporation, a constant parameter of the method, $\mathcal{A}$ is the set of arcs (edges or connections) that fully connects the set of nodes and

$\Delta\tau^{k}_{i,j}(t)$ is the amount of pheromone ant k deposits on the connections it has visited through its tour T^{k}, typically given by

$$\Delta\tau^{k}_{i,j} = \begin{cases} \dfrac{1}{L(\mathsf{T}^{k})} & \text{if connection } (i,j) \text{ belongs to } \mathsf{T}^{k} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

The coefficient ρ must be set to a value <1 to avoid unlimited accumulation of trail [39]. In general, connections that are used by many ants and which are parts of short tours, receive more pheromone and are therefore more likely to be chosen by ants in future iterations of the algorithm. A pseudo code of the ACO procedure is given in the following Figure 3.

Set ACO parameters α, β, ρ

Initialize pheromone trails matrix $\boldsymbol{\tau}$ ($N\times N$)

Repeat

Place m ants randomly on the N nodes

For i=1 to m

For j=1 to N-1

Assign probabilities for every feasible connection according to Eq. (10)

Update ant's position

End

The ant returns to its initial place, closing the tour

End

Update pheromone for each connection i,j according to Eqs. **(12)** and **(13)**

While termination criterion not satisfied

Fig. 3 Flowchart of the ACO algorithm

5 Case Studies

In order to assess the performance of the formulation of the problem defined in Section 2 along with the optimization algorithms considered, two case studies are examined: an 'academic' and a real world case study.

5.1 Academic Case Study

The first case study corresponds to an area/city having a rectangular layout composed of 8×8=64 structural blocks, while the centres of adjacent building blocks forgo 100 meters. This case study has been considered in order to calibrate the optimization algorithms used for solving the two step optimization problem, and to also assess the performance in a similar to the real world case study but with a known solution.

In the first step, the optimal assignment problem as defined in Eq. (2) is solved by means of the Particle Swarm Optimization algorithm (optimal allocation of inspection crews to city blocks). The parameters of the algorithm are: NP=50, t_{max}=500, n=64, c_1=2.0, c_2=2.0, w_{start}=0.95 (velocity weight at the beginning), w_{end}=0.5 (velocity weight at the end of the PSO iterations).

In order to validate the performance of the algorithm, two and four inspection crews have been considered. Figures 4a and 4b depict the solutions obtained for the optimum assignment problem for two and four inspection crews, respectively.

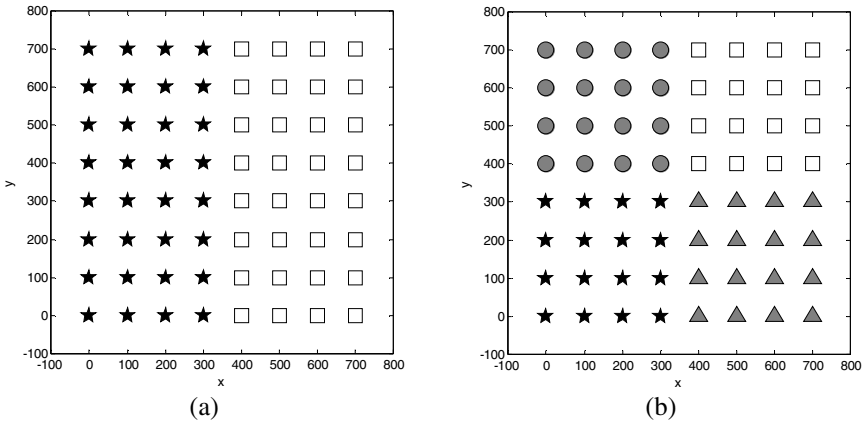
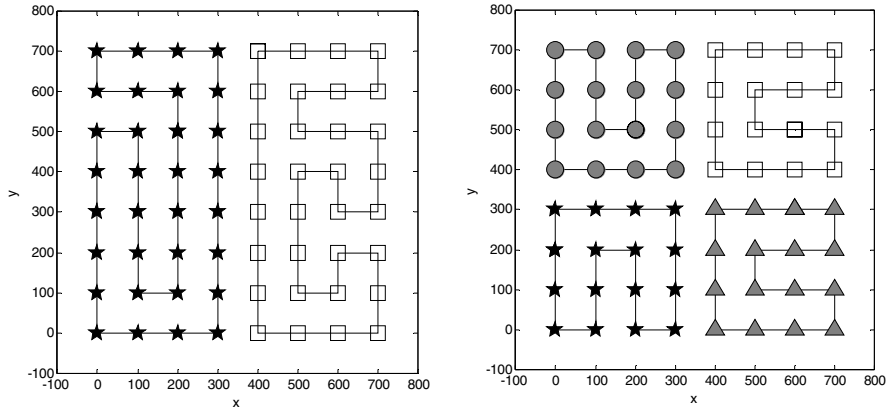


Fig. 4 Academic case study - Subdivision into structural blocks (a) two and (b) four inspection crews

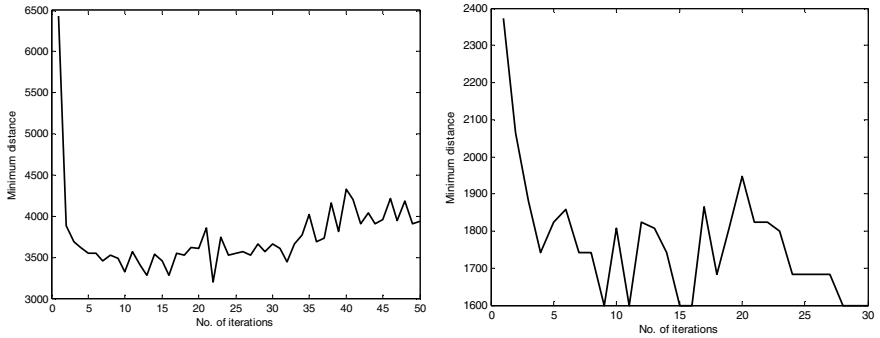
For the two inspection groups v_{max}=1 while for the four inspection groups v_{max}=3. For the solution of the assignment problem, the area and structural percentage are the same in all structural blocks, thus the solution of this problem is reduced into a problem of minimizing the distance between the centres of the structural blocks assigned to an inspection crew from the global centre of the structural blocks group. As can be seen, the optimal allocations match exactly the expected assignment of the structural blocks on the inspection groups both for the case that N_{IG}=2 and the case that N_{IG}=4.

In the second step, the inspection prioritization problem defined in Eq. (4) is solved by means of the Ant Colony Optimization algorithm. This step assigns inspection priorities – within the building blocks determined in Step 1 – for inspection groups, i.e. the first building to be inspected, the second, and so on. The parameters of the method are: *evap_rate*=0.1 (rate of pheromone evaporation), *a*=1, *b*=5, iterations were set to 50 while the number of ants was set to 150. Figures 5a and 5b depict the optimal routes achieved that correspond to the least time consuming route required for each inspection crew starting from a base (the base is the same for every inspection group).



(a) (b)

Fig. 5 Academic case study – Best route (a) two and (b) four inspection crews



(a) (b)

Fig. 6 Academic case study – Optimization history of the last group (a) for the case of two and (b) the case of four inspection crews

Figure 6 depicts the convergence histories of the ACO algorithm. The vertical axis is the minimum distance path among the ants for every iteration.

To examine the advantages of the solution obtained for the formulation of the TSP problem two alternative formulations were examined: (i) Random route selection and (ii) Closest available node. In the first strategy, an agent selects a block randomly, from the available blocks that have not yet been visited. In the second strategy, an agent selects the block that is closer to his current position, from the available blocks that have not yet been visited. If more two or more blocks are equally close, then a random selection is done.

For both solutions, 10000 simulations were examined and the average distance was compared to those obtained by the optimizers. Figure 7 depicts a randomly selected solution for the two cases. In the first strategy, the average distance was 10468 which is an increase of 227% compared to the optimal 3200 distance, while the average distance for the second strategy was 3480 which is an increase of 9% compared to the optimal distance.

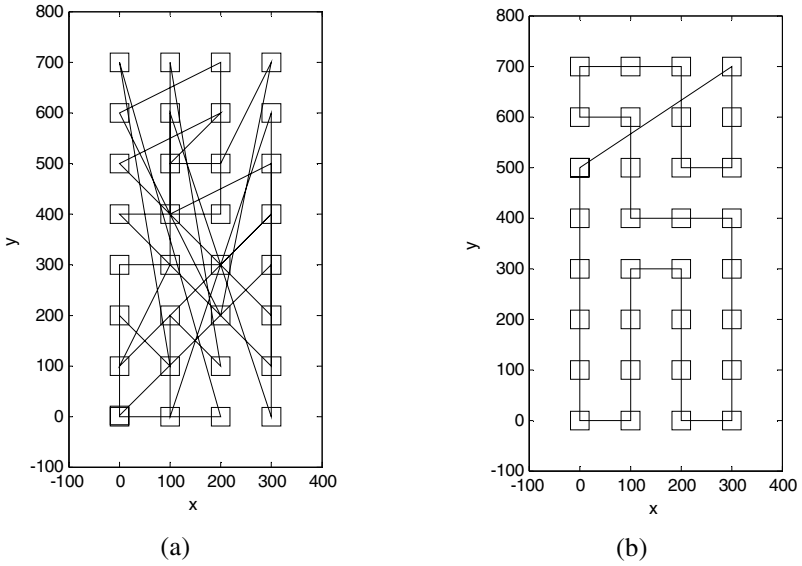


Fig. 7 Academic case study – Two solutions for the TSP problem (a) Random route selection and (b) Closest available node

5.2 Real World Case Study

The second test case corresponds to a real world case study, the city of Patras in Greece, which was considered in order to define both the problem of the inspection assignment and the inspection prioritization. The city of Patras is

decomposed into 112 structural blocks having different areas and built-up percentages, while two different sets of inspection groups (crews of inspectors) are considered. The subdivision of the city of Patras into 112 structural blocks can be seen in Figure 8a.

Two different scenarios were considered with respect to the damage level encountered on the structures due to a strong earthquake. In the first, the damages are the same in all city blocks, while in the second four areas with differential structural damage levels are considered: (i) Level 0 – no damages, (ii) Level 1 – slight damages, (iii) Level 2 – moderate damages and (iv) Level 3 – extensive damages. The four areas are shown in Figure 8b.

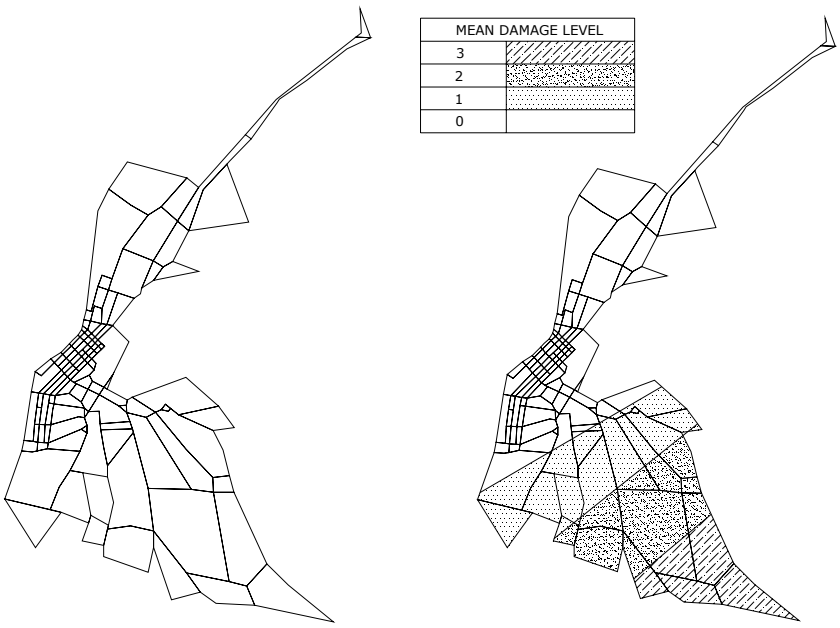
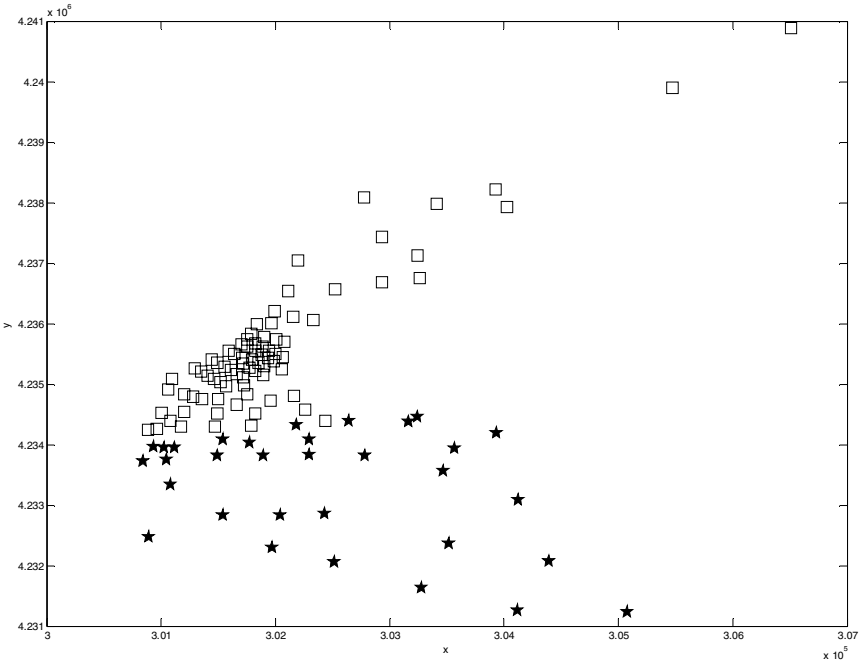


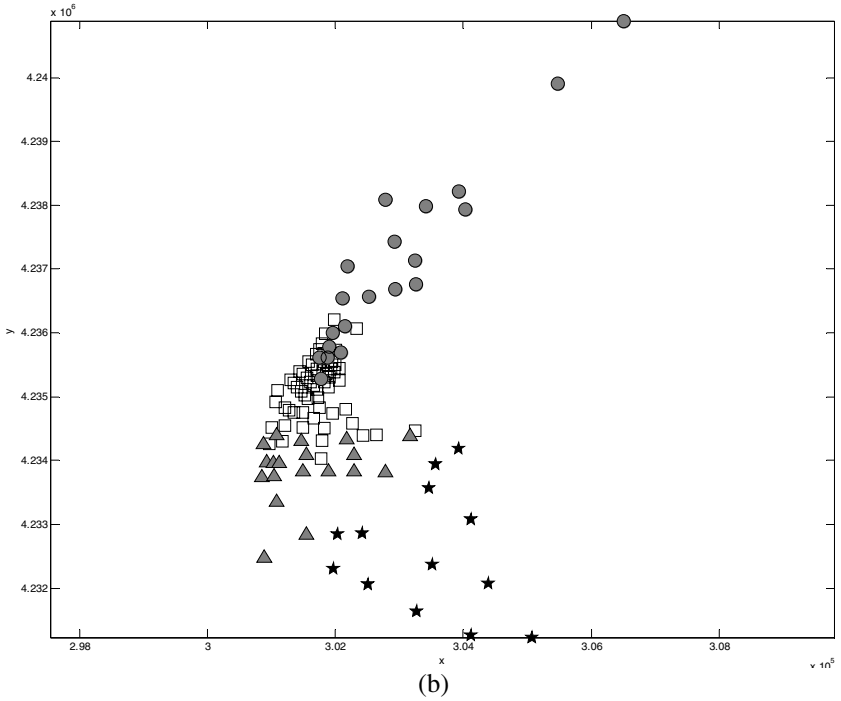
Fig. 8 City of Patras – (a) Subdivision into structural blocks and (b) Mean damage level distributed over the structural blocks

5.2.1 Uniform Distribution of Damages

In the first part of this case study, a uniform distribution of damages is examined. Similar to the previous test example, two and four inspection crews were examined. Figures 9a and 9b depict the solutions obtained for the optimal allocation problem for the two different number of inspection crews. In contrary to the academic test example the area and built-up percentages are not the same in the structural blocks.



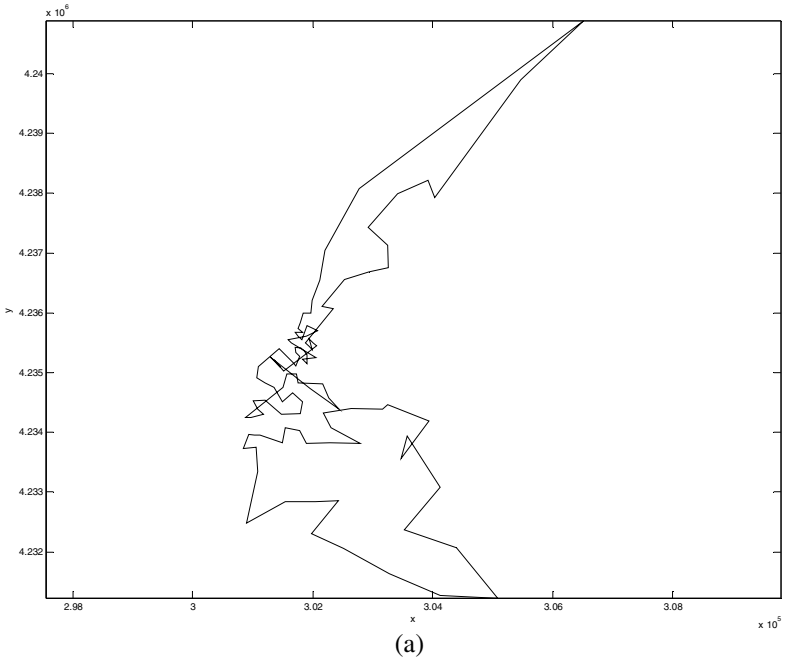
(a)



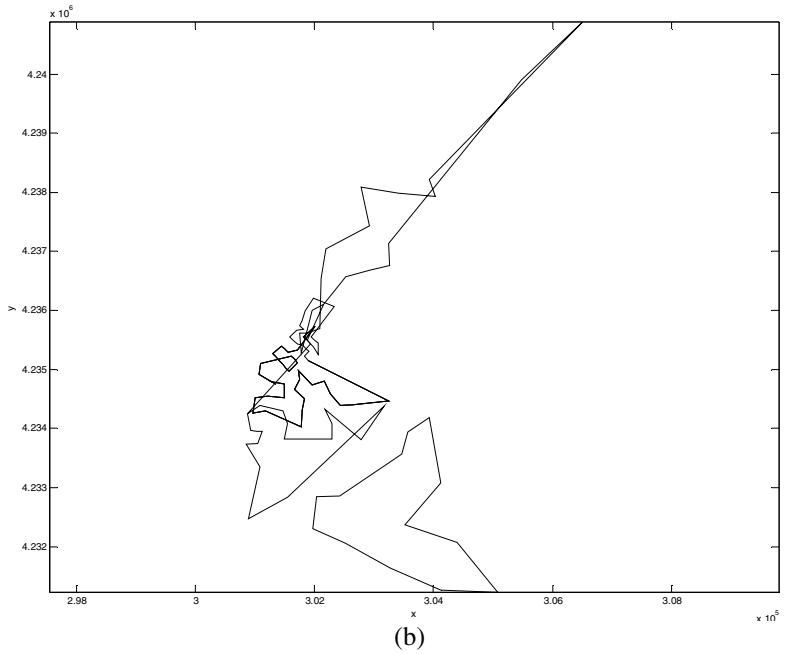
(b)

Fig. 9 City of Patras - Subdivision into structural blocks (a) two and (b) four inspection crews

In the second step, the inspection prioritization problem defined in Eq. (4) is also solved by means of the Ant Colony Optimization algorithm. Figures 10a and 10b



(a)

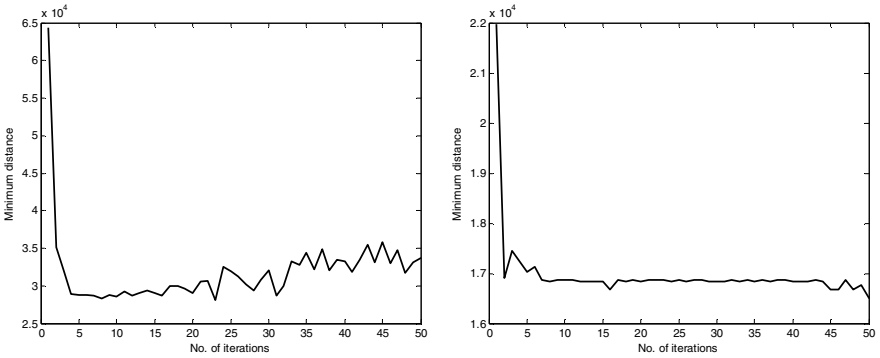


(b)

Fig. 10 City of Patras – Best route (a) two and (b) four inspection groups

depict the optimal routes achieved, corresponding to the least time consuming route required for each inspection crew departing from their base (the base is the same for every inspection crew).

The distances for the first and second group are 17444 and 28145 respectively for the two inspection groups while for the four are 10431, 12986, 9161 and 16498. Figure 11 depicts the convergence histories of the ACO algorithm. The vertical axis is the minimum distance path among the ants for every iteration.



(a) (b)

Fig. 11 City of Patras – Optimization history of the last group (a) for the case of two and (b) the case of four inspection groups

5.2.2 Non-uniform Distribution of the Damages

In the second part, a non-uniform distribution of damages is examined. The mean damage level for each region is shown in Figure 8b. Damages are assumed to follow the Gaussian distribution with mean value 0, 1, 2 and 3 for the four zones of Figure 8b. The final distribution of damages over the structural blocks can be seen in Figure 12, where a big circle denotes severe damage. In order to account for the influence of the distribution of the damages in the city's regions, the formulation of the optimal assignment problem given in Eq. (2) is modified as follows

$$\min \sum_{i=1}^{N_{IG}} \sum_{k=1}^{n_{SB}^{(i)}} \left[d(SB_k, C_i) \cdot D(k) \cdot DF(k) \right] \tag{14}$$

where $DF(k)$ is the damage factor corresponding to each damage level, as shown in Table 3. Figures 13a and 13b depict the solutions obtained for the optimum allocation problem for the two different number of inspection crews.

Table 3 Damage Factor (DF) corresponding to each damage level

Damage level	Damage Factor (DF)
0	1.0
1	1.2
2	1.5
3	2.0

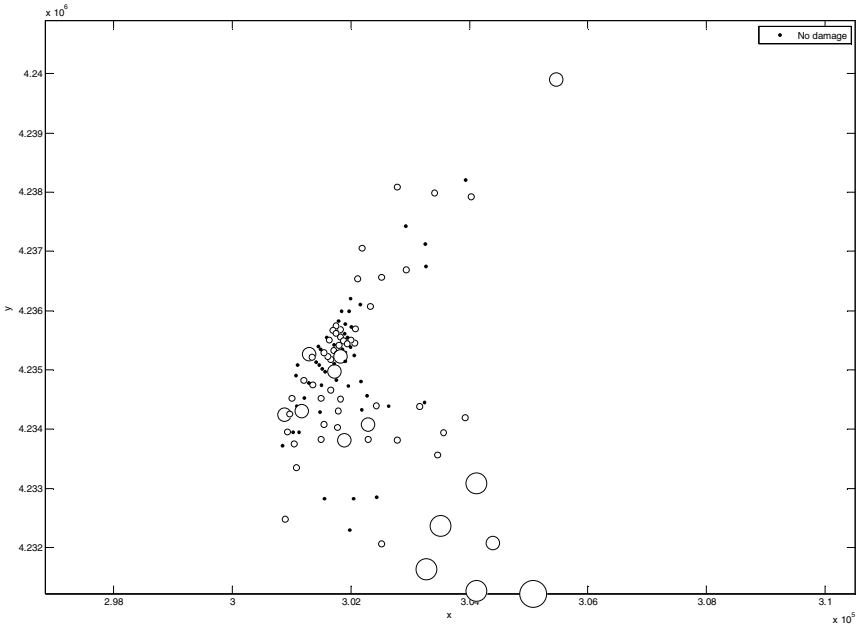
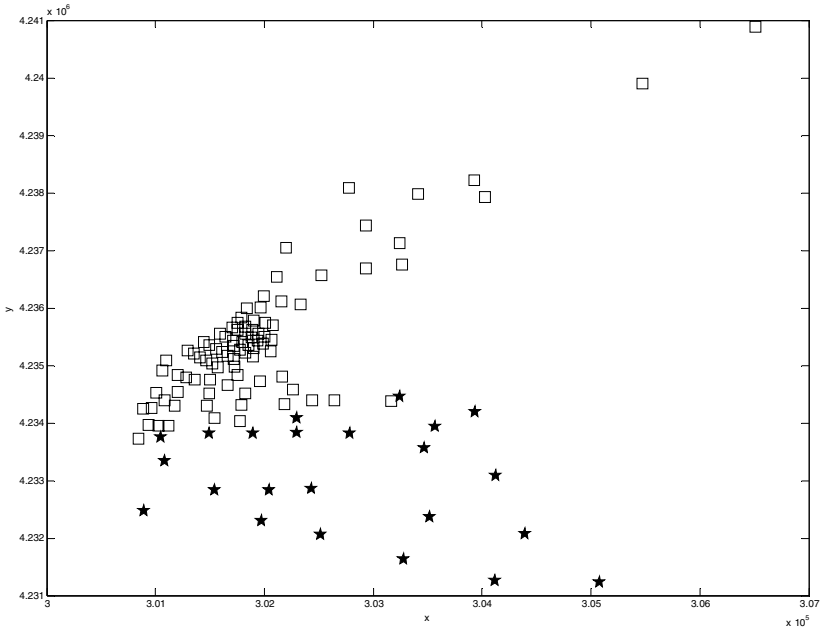
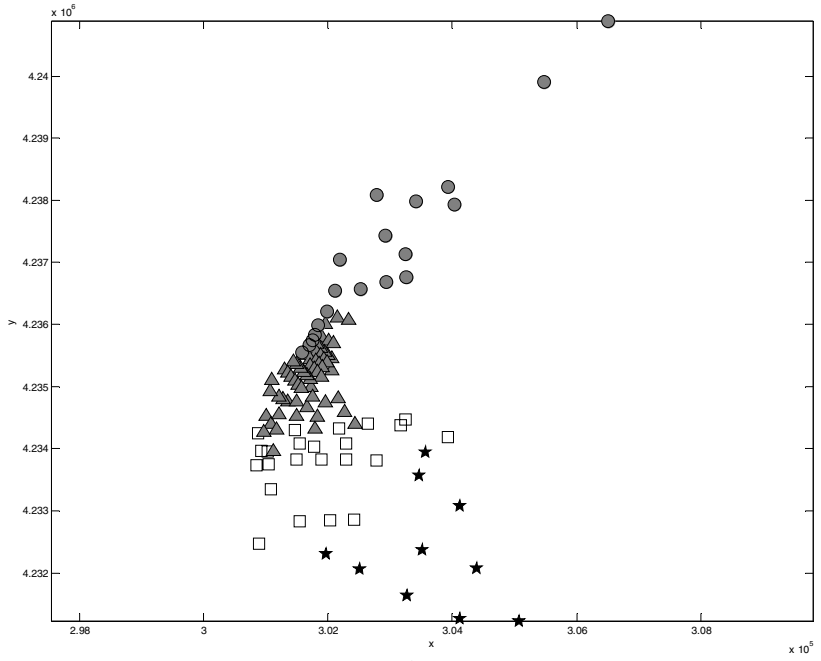


Fig. 12 City of Patras – Distribution of the damage levels

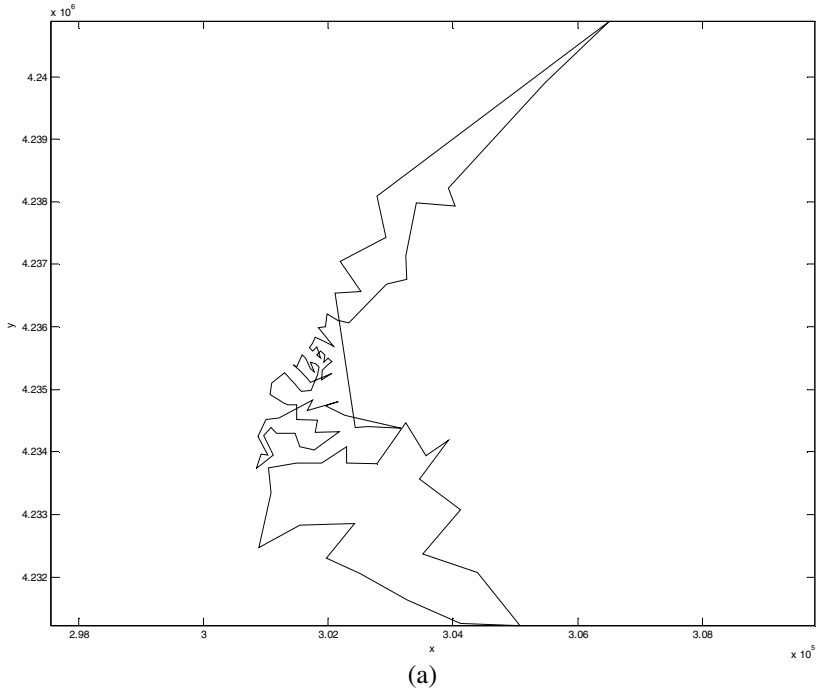


(a)

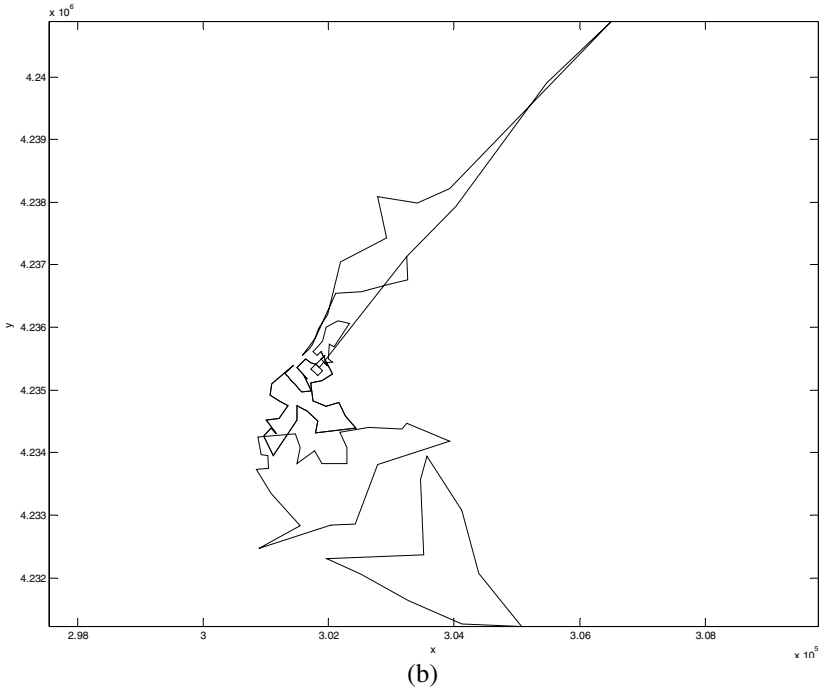


(b)

Fig. 13 City of Patras - Subdivision into structural blocks (a) two and (b) four inspection crews



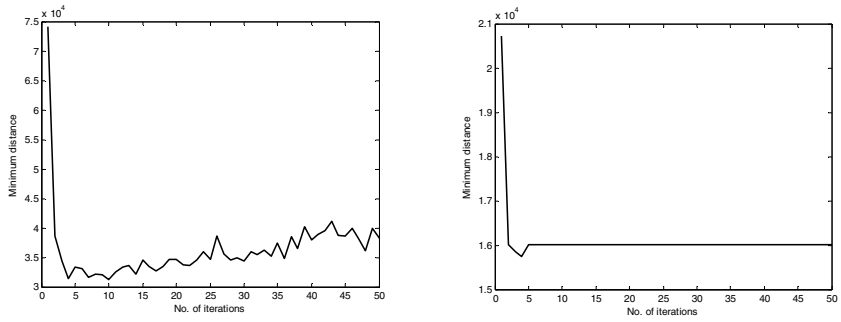
(a)



(b)

Fig. 14 City of Patras – Best route (a) two and (b) four inspection crews

In the second step, the inspection prioritization problem defined in Eq. (4) is solved by means of the Ant Colony Optimization algorithm. Figures 14a and 14b depicts the optimum routes achieved, corresponding to the less time consuming route required for each inspection group imitating from their base. The base is the same for every inspection crew. The distances for the first and second group are 17121 and 31540 respectively for the two inspection groups while for the four are 9633.7, 10939, 11383 and 15740.



(a) (b)

Fig. 15 City of Patras – Optimization history of the last group (a) for the case of two and (b) the case of four inspection crews

Figure 15 depicts the convergence histories of the ACO algorithm. The vertical axis is the minimum distance path among the ants for every iteration.

6 Conclusions

Following a natural hazard, the condition of the critical infrastructures must be assessed and damages have to be identified. Inspections are therefore necessary, immediately after the catastrophic event, since failure to quickly inspect, repair and/or rehabilitate the infrastructure system, particularly in densely populated metropolitan regions, might delay search and rescue operations and relief efforts, which increases the suffering of the survivors. Specialized crews must be dispatched and inspect critical infrastructures. The objective of the current work was to schedule critical infrastructures inspection crews following an earthquake in densely populated metropolitan regions. In this chapter two formulations have been successfully implemented: in the first, the structural blocks are assigned to different inspection groups with an effort to equally distribute the workload between the groups, while in the second the optimal route for each group was determined with an effort to minimize the distance that each inspection group has to cover. A Particle Swarm Optimization and an Ant Colony Optimization-based framework were implemented for dealing with the problem at hand and they both resulted in tractable and rapid response models.

References

[1] Altay, N., Greene, W.G.: OR/MS research in disaster operations management. European Journal of Operational Research 175, 475–493 (2006)
[2] Dong, W.M., Chiang, W.L., Shah, H.C.: Fuzzy information processing in seismic hazard analysis and decision making. Soil Dynamics and Earthquake Engineering 6(4), 2202–2226 (1987)
[3] Peizhuangm, W., Xihui, L., Sanchez, E.: Set-valued statistics and its application to earthquake engineering. Fuzzy Sets and Systems 18(3), 347–356 (1986)
[4] Tamura, H., Yamamoto, K., Tomiyama, S., Hatono, I.: Modeling and analysis of decision making problem for mitigating natural disaster risks. European Journal of Operational Research 122(2), 461–468 (2000)
[5] Mendonca, D., Beroggi, G.E.G., Wallace, W.A.: Decidion supprt for improvisation during emergency response operations. International Journal of Emergency Management 1(1), 30–38 (2001)
[6] Mendonca, D., Beroggi, G.E.G., van Gent, D., Wallace, W.A.: Designing gaming simulations for the assessment of Group decision support Systems in emergency response. Safety Science 44, 523–535 (2006)
[7] Viswanath, K., Peeta, S.: Multicommodity maximal covering network design problem for planning critical routes for earthquake response. Transportation Research Record 1857, 1–10 (2003)
[8] Nicholson, A., Du, Z.-P.: Degradable transportation networks systems: An integrated equilibrium model. Transportation Research part B 31(3), 209–223 (1997)
[9] Sakakibara, H., Kajitani, Y., Okada, N.: Road network robustness for avoiding functional isolation in disasters. ASCE Journal of Transportation Engineering 130(5), 560–567 (2004)
[10] Sohn, J.: Evaluating the significance of highway network links under the flood damage: An accessibility approach. Transportation Research part A 40(6), 491–506 (2006)
[11] Song, J., Kim, T.J., Hewings, G.J.D., Lee, J.S., Jang, S.-G.: Retrofit priority of transport network links under an earthquake. ASCE Journal of Urban Planning and Development 129(4), 195–210 (2003)
[12] Verter, V., Lapierre, S.: Location of preventive healthcare facilities. Annals of Operations Research 110, 123–132 (2002)
[13] Barbarosoglou, G., Arda, Y.: A two-stage stochastic programming framework for transportation planning in disaster response. Journal of the Operational Research Society 55(1), 43–53 (2004)
[14] Barbarosoglou, G., Ozdamar, L., Cevik, A.: An Interactive approach for hierarchical analysis of helicopter logistics in disaster relief operations. European Journal of Operational Research Society 140(1), 118–133 (2002)
[15] Fiedrich, F., Gehbauer, F., Rickers, U.: Optimized resource allocation for emergency response after earthquake disasters. Safety Science 35(1-3), 41–57 (2000)
[16] Ozdamar, L., Ekinci, E., Kucukyazici, B.: Emergency logistics planning in natural disasters. Annals of Operations Research 129(1-4), 217–245 (2004)
[17] Bell, M.G.H.: A game theory approach to measuring the performance reliability of transportation networks. Transportation Research part B 34(6), 533–545 (2000)

[18] Chang, S.E., Nojima, N.: Measuring post-disaster transportation system performance: The 1995 Kobe earthquake in comparative perspective. Transportation Research part A 35(6), 475–494 (2001)
[19] Karaouchi, F., Lida, Y., Shimada, H.: Evaluation of road network reliability considering traffic regulation after a disaster. In: Bell, M.G.H. (ed.) The Network Reliability of Transport: Proceedings of the 1st International Symposium on Transportation Network Reliability (INSTR). Elsevier, Oxford (2001)
[20] Li, Y., Tsukaguchi, H.: Improving the reliability of street networks in highly densely populated urban areas. In: Bell, M.G.H., Lida, Y. (eds.) The Network Reliability of Transport: Proceedings of the 1st International Symposium on Transportation Network Reliability (INSTR). Elsevier, Oxford (2001)
[21] Cret, L., Yamakazi, F., Nagata, S., Katayama, T.: Earthquake damage estimation and decision-analysis for emergency shutoff of city gas networks using fuzzy set theory. Structural Safety 12(1), 1–19 (1993)
[22] Song, B., Hao, S., Murakami, S., Sadohara, S.: Comprehensive evaluation method on earthquake damage using fuzzy theory. ASCE Journal of Urban Planning and Development 122(1), 1–17 (1996)
[23] Karlaftis, M.G., Kepaptsoglou, K.L., Lampropoulos, S.: Fund allocation for transportation network recovery following natural disasters. ASCE Journal of Urban Planning and Development 133(1), 82–89 (2007)
[24] Colorni, A., Dorigo, M., Maniezzo, V.: Distributed Optimization by Ant Colonies. In: Varela, F., Bourgine, P. (eds.) Proceedings of the First European Conference on Artificial Life, pp. 134–142. Elsevier Publishing, Paris (1992)
[25] Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., Shmoys, D.B.: The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization. Wiley, New York (1985)
[26] Kennedy, J., Eberhart, R.: Particle swarm optimization. In: IEEE Int. Conf. on Neural Networks, Piscataway, NJ, USA, vol. IV, pp. 1942–1948 (1995)
[27] Perez, R.E., Behdinan, K.: Particle swarm approach for structural design optimization. Computers and Structures 85, 1579–1588 (2007)
[28] Bochenek, B., Foryś, P.: Structural optimization for post-buckling behavior using particle swarms. Structural and Multidisciplinary Optimization 32(6), 521–531 (2006)
[29] He, Q., Wang, L.: An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Engineering Applications of Artificial Intelligence 20(1), 89–99 (2007)
[30] Liang, J.J., Suganthan, P.N.: Dynamic Multi-Swarm Particle Swarm Optimizer with a Novel Constraint-Handling Mechanism. In: IEEE Congress on Evolutionary Computation, CEC 2006, pp. 9–16 (2006)
[31] Mezura-Montes, E., Lopez-Ramirez, B.C.: Comparing bio-inspired algorithms in constrained optimization problems. In: IEEE Congress on Evolutionary Computation, CEC 2007, pp. 662–669 (2007)
[32] Munoz-Zavala, A.E., et al.: PESO+ for Constrained Optimization. In: IEEE Congress on Evolutionary Computation, 2006, pp. 231–238 (2006)
[33] Ye, D., Chen, Z., Liao, J.: A New Algorithm for Minimum Attribute Reduction Based on Binary Particle Swarm Optimization with Vaccination. In: Advances in Knowledge Discovery and Data Mining, pp. 1029–1036 (2007)
[34] Hassan, R., et al.: A Comparison of Particle Swarm Optimization and the Genetic Algorithm. In: Proceedings of the 46th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference, Austin, Texas, USA (2005)

[35] Angeline, P.J.: Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 601–610. Springer, Heidelberg (1998)
[36] Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: IEEE World Congress on Computational Intelligence, Anchorage, AK, USA, pp. 69–73 (1998)
[37] Dorigo, M.: Optimization, Learning and Natural Algorithms. Politecnico di Milano, Milano (1992)
[38] Dorigo, M., Stützle, T.: Ant Colony Optimization. The MIT Press, Cambridge (2004)
[39] Colorni, A., Dorigo, M., Maniezzo, V.: An Investigation of Some Properties of an Ant Algorithm. In: Manner, R., Manderick, B. (eds.) Proceedings of the Parallel Problem Solving from Nature Conference (PPSN 1992), pp. 509–520. Elsevier Publishing, Brussels (1992)