mathematics

Article

Social Network Optimization for WSN Routing: Analysis on Problem Codification Techniques

Alessandro Niccolai *, Francesco Grimaccia, Marco Mussetta, Alessandro Gandelli and Riccardo Zich

Dipartimento di Energia, Politecnico di Milano,Via Lambruschini 4, 20156 Milano, Italy; francesco.grimaccia@polimi.it (F.G.); marco.mussetta@polimi.it (M.M.); alessandro.gandelli@polimi.it (A.G.); riccardo.zich@polimi.it (R.Z.)

* Correspondence: alessandro.niccolai@polimi.it

Received: 25 February 2020; Accepted: 9 April 2020; Published: 14 April 2020

Abstract: The correct design of a Wireless Sensor Network (WSN) is a very important task because it can highly influence its installation and operational costs. An important aspect that should be addressed with WSN is the routing definition in multi-hop networks. This problem is faced with different methods in the literature, and here it is managed with a recently developed swarm intelligence algorithm called Social Network Optimization (SNO). In this paper, the routing definition in WSN is approached with two different problem codifications and solved with SNO and Particle Swarm Optimization. The first codification allows the optimization algorithm more degrees of freedom, resulting in a slower and in many cases sub-optimal solution. The second codification reduces the degrees of freedom, speeding significantly the optimization process and blocking in some cases the convergence toward the real best network configuration.

Keywords: wireless sensor networks; routing; Swarm Intelligence; Particle Swarm Optimization; Social Network Optimization

1. Introduction

The Internet of Things paradigm is increasing the importance of Wireless Sensor Networks (WSN) in which a set of small simple sensors are interconnected for creating a very complex structure. The information sensed by all the nodes of the network should be sent to the cluster head that processes them and exploits the information [1].

The design of a WSN arises some issues in terms of deployment and operational costs. In this framework, Evolutionary Optimization Algorithms (EAs) can be very important tools for the system design because of their flexibility and ease of use.

The research on EAs has two main preferential directions: on the one hand, more complex operators are introduced and analyzed in the algorithms for improving their performance [2]; on the other hand, their field of applicability is analyzed and enlarged [3].

The most used EAs are Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) [4]. PSO and Ant Colony Optimization (ACO) [5] are the most common algorithms that belong to Swarm Intelligence.

Among EAs, Differential Evolution (DE) algorithm has been widely applied and studied [6,7]. This algorithm is not-biologically inspired; while it was originally designed for real-value problems [8], it has been implemented also for discrete problems [9]. This algorithm has been applied to a wide range of multimodal problems, such as neural network training [10]. Biogeography-Based Optimization (BBO) is a more recently developed EA based on the survival mechanism of species in different environments [11].

The design of a WSN is a very challenging task: in fact, several problems can be faced and should be solved for reducing the operational cost of them [12]. The first problem that can be faced in the design of a WSN is the deployment of sensors in the sensing field: the typical objectives are the coverage maximization. Another important problem in WSN is clustering the sensors for reducing the number of cluster heads, for reducing the information delay, and for reducing the energy consumption. This problem has been widely addressed with Evolutionary Algorithms [13] and Swarm Intelligence Algorithms [14].

With recent advancements in massive parallel computing technologies, the problem of scheduling resources and tasks in WSN is becoming more and more important. The topic of task allocation is faced with evolutionary algorithms in [15], where the task scheduling in heterogeneous distributed systems is evaluated comparing a multi-objective evolutionary algorithm with a hybrid GA, and, in [16], where GA is applied for a similar purpose, performing tests on a Java scheduler with a homogeneous set of processors. More recently, in [17], a Logic Based evolutionary algorithm compared to a binary PSO is applied to task allocation in wireless sensor networks. In [18], Social Network Optimization (SNO) is used for solving the task allocation problem.

Among the different techniques that can be used in WSN design and operation optimization, Swarm Intelligence algorithms have been extensively adopted. In [19], an algorithm based on ACO is exploited to find the optimal information path from a source node to a sink node in a multi-hop WSN. ACO is also applied in [20] for the path design of a mobile data collection node.

Lifetime maximization is a critical issue in WSN, especially when the sensors are deployed in a field in which maintenance is very difficult. In particular, this problem can be managed in different ways.

In [21], the authors proposed an enhanced Hybrid Genetic Algorithm for maximizing the WSN lifetime by means of an optimal scheduling of disjoint set of sensors.

The lifetime maximization is achieved in [22] by selecting the proper routing in the WSN by means of Genetical Swarm Optimization. The same optimization algorithm is implemented in [23], in which a real encoding of the chromosome is adopted.

In [24], Particle Swarm Optimization algorithm is used for a multi-objective problem in which the network lifetime maximization achieved by a proper selection of the routing is combined with the quality of service maximization.

From the analysis of all the presented papers, it is clear that the problem codification, i.e., how the network configurations are codified in the optimization variables, is a key aspect for the applicability of EAs in the IoT framework: in fact, the possible network configurations significantly increases with the number of sensors involved. A proper selection of the codification method can reduce the computational time required for the optimization: in particular, it can reduce the size of the search space and the number of unfeasible solutions. On the other hand, the reduction of the search space can be detrimental because it can eliminate the optimal solution.

In this paper, two problem codifications are analyzed: in the first one, a basic heuristic logic is used for reducing the search space size without losing degrees of freedom. In the second codification, a more complex heuristic is used for completely avoiding unfeasible solutions, but the degrees of freedom are highly reduced.

The problem of routing was solved with both the codifications on 27 different networks. Two swarm intelligence optimization algorithms were tested: SNO and PSO. The objective of this design problem is the maximization of the network lifetime, in which the transmitting and receiving energy requirements are considered.

The paper is structured as follows. Section 2 provides a brief description of PSO and SNO. Section 3 contains the description of the adopted WSN model, the two problem codification techniques, and the definition of the problem environment. In Section 4, the obtained results are presented and discussed. Finally, in Section 5, some conclusions are drawn.

2. Swarm Intelligence

Swarm Intelligence Algorithms are an important class of Evolutionary Optimization Algorithms. In this paper, Particle Swarm Optimization and Social Network Optimization are used. In the following, a brief description of both these algorithms is presented.

2.1. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a milestone in Swarm Intelligence algorithms [25]. Its operators are derived from the concept of collective intelligence, which can be summarized in the following three sentences [26]:

- Inertia: 'I continue on my way'.
- Influence of the past: 'I've always done in this way'.
- Influence by society: 'If it worked for them, it would also work for me'.

Each individual of the population represents a particle, which is characterized by a position and velocity.

At each iteration of the algorithm, the personal best position found by each particle (PB) and the global best (GB) found by the entire swarm are computed. These points are the attractors of each individual. In particular, the velocity is updated in the following way:

$$\mathbf{v}_i(t+1) = \omega \mathbf{v}_i(t) + c_1(\mathbf{PB}_i - \mathbf{p}_i(t)) + c_2(\mathbf{GB} - \mathbf{p}_i(t)) \tag{1}$$

where $\mathbf{v}_i$ is the velocity of the ith particle and $\mathbf{p}_i$ its position in the search space. $\mathbf{PB}_i$ is the personal best while $\mathbf{GB}$ is the global best. Finally, ω, c_1, and c_2 are three user defined parameters.

At each iteration, the position is updated in the following way:

$$\mathbf{p}_i(t+1) = \mathbf{p}_i(t) + \mathbf{v}_i(t+1) \tag{2}$$

PSO performance are greatly influenced by the selection of the algorithm working parameters. The implemented PSO version is characterized by the following parameters: the inertia weight (ω), the personal learning coefficient (c_1), the global learning coefficient (c_2), and the velocity clamping (V_{rsm}) [27,28]. Moreover, the algorithm performance depends on the population size (n_{pop}) that is related to the number of iterations (n_{iter}) and the number of objective function calls (n_{call}) by the following equation:

$$n_{pop} \cdot n_{iter} = n_{call} \tag{3}$$

2.2. Social Network Optimization

Social Network Optimization is a recently developed population-based, Swarm Intelligence algorithm that takes its inspiration from the information sharing process in Online Social Network [29].

The population of the algorithm is composed by *users* of the social network who interact online by publishing some *posts*. In fact, at each iteration, the users express their opinions by means of the *status* contained in the post. In addition to this information, the post contains the name of the user that have posted it, the time at which it is posted, and a visibility value. The process of passing from *opinions* to a post *status* is called *linguistic transposition*.

The status corresponds to the candidate solution of the optimization problem, while the visibility value is created by means of a proper mapping of the cost function associated to the specific candidate solution, as shown in Figure 1.

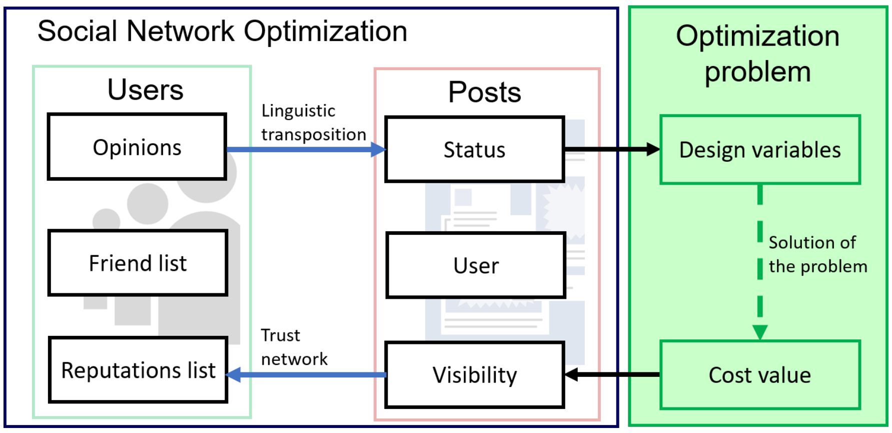


Figure 1. Composition and interaction of data structures in SNO.

The visibility values of the entire population are used to update the reputations of all the users of the social network. This operation transfers global information among the entire population. The reputations are used for creating one of the two interaction structures of SNO: the trust network.

The second interaction network is the friend network. These two networks are very different one from each other: the friend network leads to more consistent interactions, its variability is lower, and its modifications are related to out-of-the-social elements. The trust network creates weaker connections, it varies quickly, and it is modified according to the visibility values, as shown in Figure 2.

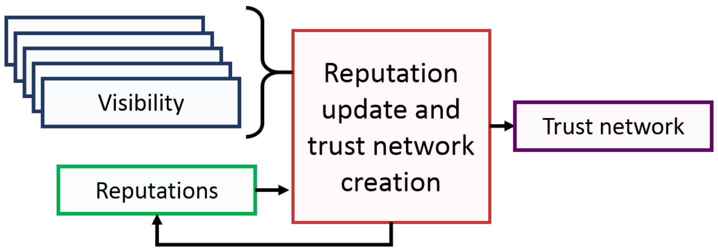


Figure 2. Schematization of the process of reputation update and creation of trust network.

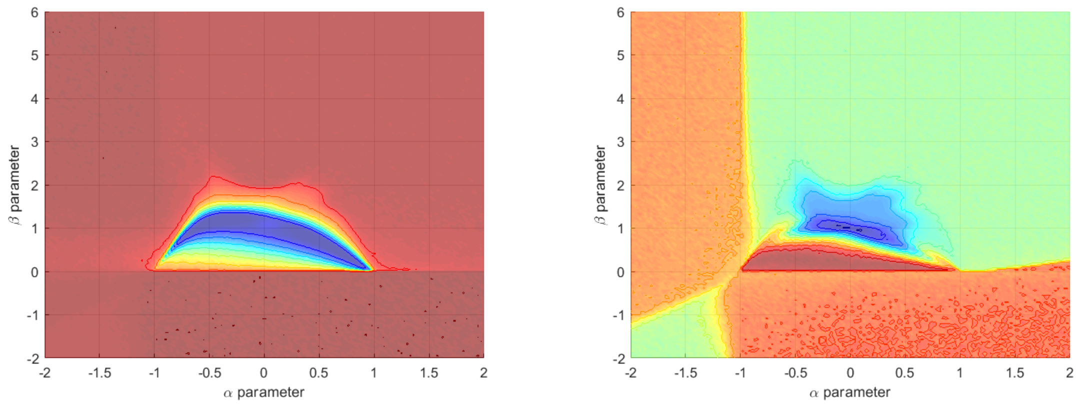
From the interaction networks, each user extracts some ideas that compose the attraction point that is used for creating the new opinions. The implemented operator emulates the assumption of a complex contagion, which guarantees a better trade-off between exploration of the domain and the exploitation of the acquired knowledge:

$$\mathbf{o}_u(t+1) = \mathbf{o}_u(t) + \alpha[\mathbf{o}_u(t) - \mathbf{o}_u(t-1)] + \beta[\mathbf{a}_u(t) - \mathbf{o}_u(t)] \tag{4}$$

All these operators make SNO a quite complex algorithm, but they give to it the possibility to work very well in different kinds of problems and they reduce the risk of stagnation in local minima, which is a well-known issue of PSO.

The most important algorithm parameters are α and β because they tune the effective behavior of the population. Two different analyses have already been performed for properly selecting these parameters: The first one is an analytical analysis of a simplified version of the complex contagion. The second one is a numerical parametric analysis performed on different benchmarks. More details on these analyses can be found in [30].

The parametric analysis on the parameters α and β has been performed on two standard benchmarks for EAs: the Ackley and Schwefel functions. The termination criterion is set to 5000 objective function call and 50 independent trials have been done on 20D functions. Figure 3 shows the results of the parametric analysis: the color is proportional to the cost value, where red is high cost and blue low cost. The results show that the high quality solutions area is different, but it is possible to find a good set of parameters for both the functions. In particular, the selected set of parameters is $\alpha = 0.1$ and $\beta = 0.45$.



(**a**) Ackley function. (**b**) Schwefel function.

Figure 3. Parametric analysis on α and β on two different benchmarks.

In [31], it is possible to find a comparison between SNO and other EAs, in which the performance of SNO is assessed on standard benchmarks.

3. Wireless Sensor Network

A Wireless Sensor Network is composed of a set of sensors, deployed in a field, that can communicate by means of a multi-hop protocol to a central node, called the cluster head.

In the following, the adopted network model is described and then the two proposed problem codification techniques are described. Finally, the optimization environment for this problem is analyzed.

3.1. Network Energy Model

The analyzed WSN is composed by a set of equal sensor nodes deployed randomly in the space. Each of these nodes has a maximum communication distance that is imposed by the acceptable signal-to-noise ratio.

The communication between the nodes and the central node, the cluster head, happens by means of a multi-hop protocol: in this way, even nodes quite far from the cluster head can send to it the sensed information.

Each sensor but the cluster head is fed by batteries with a total capacity of 9kJ, thus the network lifetime is limited by the energy consumption of the most stressed sensor.

The energy consumption of each node can be computed as function of the transmitted and received bits: in fact, it is assumed that the sensing energy is negligible with respect to the communication energy [20]. Thus, the total energy consumption is composed by two terms:

$$E_i = E_{\text{trx},i} + E_{\text{amp},i} \tag{5}$$

where $E_{\text{trx},i}$ is the total energy required for transmission and reception of information:

$$E_{\text{trx},i} = b_{\text{trx},i} \cdot e_{\text{trx}} \tag{6}$$

where e_{trx} is the energy required to keep on the communication equipment and $b_{\text{trx},i}$ is the total number of bit transmitted and received.

The second term of the total energy consumption is the amplification energy required for the transmission of the information. It depends on the number of transmitted bits (b_{ij}) and on the transmission distance (d_{ij}) between the two nodes i and j:

$$E_{\text{amp},i,j} = E_{\text{amp}} \cdot b_{ij} \cdot d_{ij} \tag{7}$$

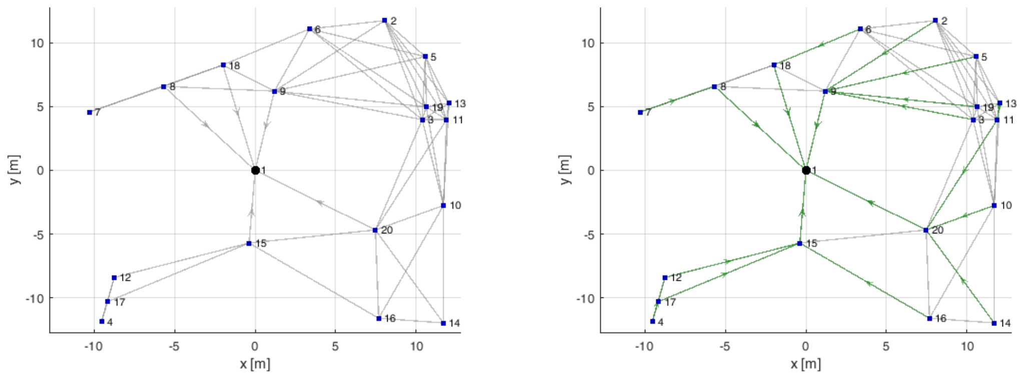
Both the sensing energy and the energy consumed in idle time are considered negligible with respect to the other terms.

The transmitted and received bits are functions of the network topology, i.e., of the interconnections between sensors and can be easily calculated.

The network is composed by a set of randomly deployed sensors: as introduced above, the feasible connections between sensors is determined by the maximum communication distance between nodes.

The feasible connections are bidirectional if they involve only sensor nodes, while are considered monodirectional if one of the two nodes is the cluster head, as shown in Figure 4a, where the directed connections are indicated with the arrow.

The feasible connections are more than the required ones, thus it is possible to select the effective final network topology by identifying the *active* connections among the feasible ones, as shown in Figure 4b.



(**a**) Feasible connections (**b**) Active connections

Figure 4. Example of a 20-node network. The feasible connections are shown in grey (**a**), while the active connections are in green in (**b**). Node 1 is the cluster head.

The selection of the active connection should be performed also selecting the direction of the communication. This is an important aspect for the problem codification, as analyzed below.

Thus, when the active connections are selected, the network can be represented by a directed graph; by means of a graph search starting from the nodes with higher depth, it is possible to determine the flow of information in the network, and thus the number of received and transmitted bits.

This network model was implemented in Matlab, adopting the parameters shown in Table 1.

Table 1. List of network parameters.

Parameter	Symbol	Value	Parameter	Symbol	Value
ine Communication package size	b_{pack}	100 bit	Communication energy	E_{TRX}	5 pJ/b
Stored energy	E_s	9 kJ	Amplification energy	E_{amp}	0.01 pJ/b

The objective of this optimization problem is the maximization of the network lifetime, i.e., the number of working cycles in which all the sensors have a residual stored energy.

The design variables of this problem are the active connections in the WSN; in fact, by changing the active connections, the number of bits received and transmitted by the sensors is modified and, thus, the sensor lifetime is changed. It is possible to codify these design variables in different ways in the optimization framework, changing the search space size and the number of unfeasible solutions. A deeper analysis of the possible codification methods is provided in the following.

To the best of the authors' knowledge, there are some papers that have proven that the problem of the lifetime maximization by properly selecting the routing paths in a WSN is a NP-hard problem [32,33].

3.2. Problem Codification

The presented model of WSN can be codified in the optimization environment in different methods that affect the number of optimization variables, their type, the possibility to find unfeasible solutions, and the search space size.

The most basic codification of this problem can be performed using binary design variables, each one representing if a feasible connection is also active. This formulation is very simple and generic, but it leads to a high number of non-connected networks, i.e., networks in which at least one node cannot communicate with the cluster head. Moreover, many solutions are surely suboptimal, such as ones with loops or with a node that transmits its information to more than one single sensors. This possibility, which can be important for reliability analysis of the communication, is not optimal in this context because the transmission and receiving energy of some nodes is doubled.

In this strategy, the number of design variables is equal to the number of feasible connections in the network, and thus generally grows more than linearly with the number of nodes.

It is possible to design more complex problem codifications by analyzing some features of optimal solutions. First, each sensor should be able to transmit its information to another one.

In this framework, it is possible to associate to each node its adjacency list (as shown in Figure 5) and select one target node from this list. With this codification, the problem is characterized by $N - 1$ integer design variables, where N is the number of nodes.

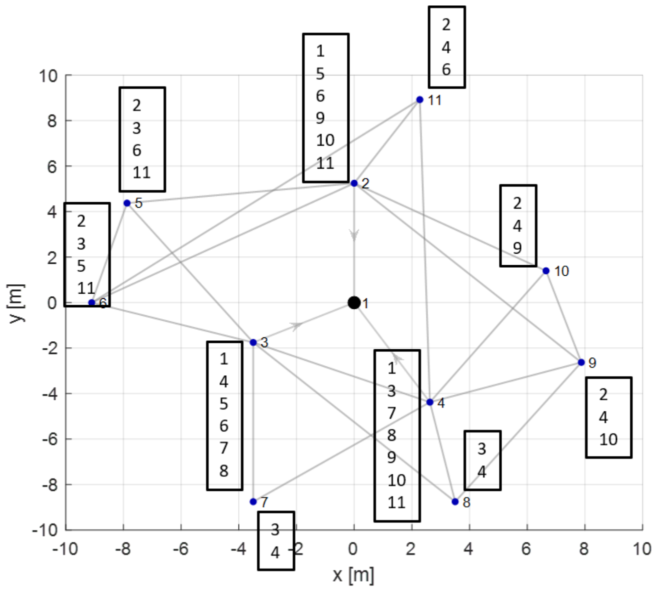
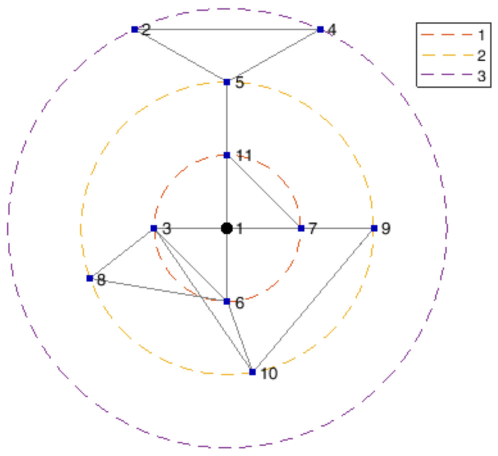


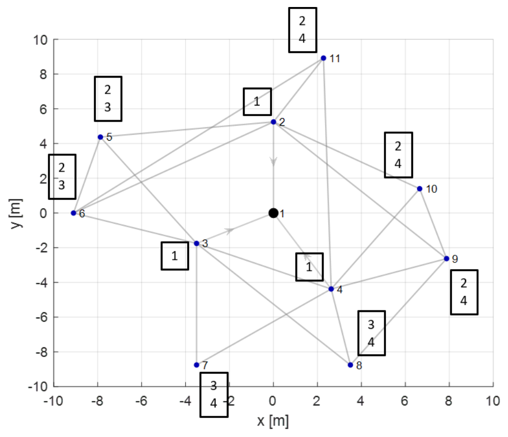
Figure 5. WSN with 11 nodes in which, for each node, its adjacency list is defined.

The drawback of this codification (called in the following adjacency-based codification) concerns the possibility of creating non-connected networks during the optimization process or looped graphs.

To avoid these drawbacks, it is possible to consider the node depth, that is the number of hops needed for communicating from the analyzed sensor to the cluster head. Figure 6a shows the same graph seen before in which the node position is related with node depth. Each circle corresponds to a different depth.



(**a**) Representation of node depth.



(**b**) Depth-based adjacency lists.

Figure 6. WSN with 11 nodes: (**a**) the node depth is underlined with the circles; and (**b**) to each node its depth-based adjacency list is defined.

For the second codification method, depth-based adjacency lists have been created: in each list, only the node with smaller node depth are included. Figure 6b shows the same network of Figure 5 in which the depth-based lists are shown. In addition, in this case, the design variables are integer numbers.

As it is possible to clearly see, the number of degrees of freedom for the algorithm is significantly reduced; this eliminates completely the possibility to have non-connected nodes in the network, but it eliminates some paths that can be optimal.

These two codification techniques make this problem combinatorial. However, in most real cases, it is impossible to test all of them due to the search space size.

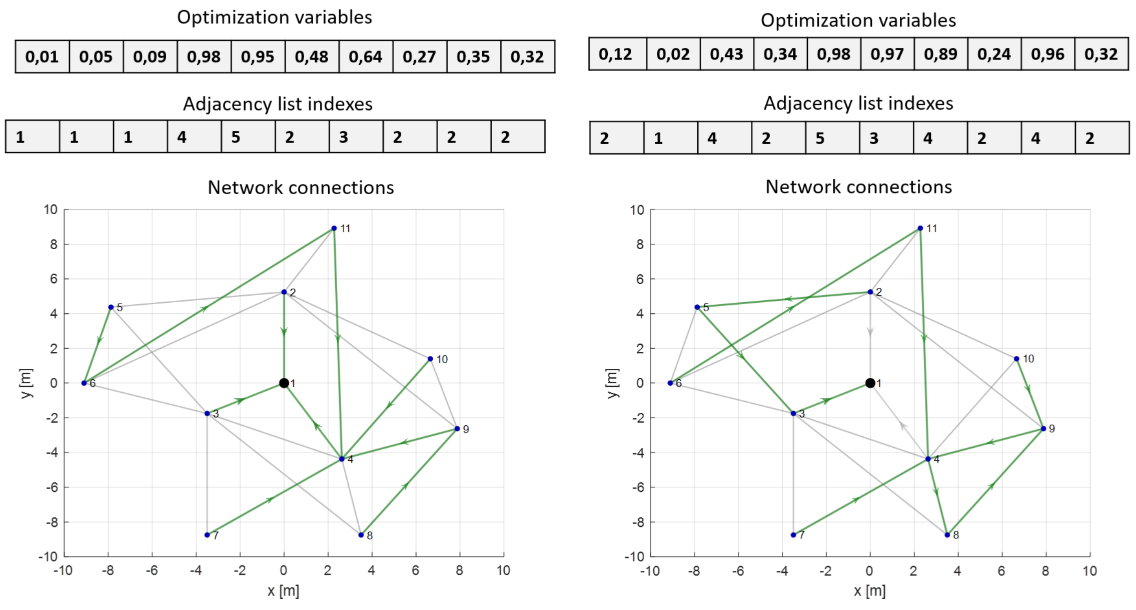
3.3. Performance Calculation

In this section, an example of how the system works is presented: it starts from the optimization variables and shows all the steps required for the calculation of the cost value.

The optimizer works with real variables that are translated into integer values by means of a set of thresholds [23]. Each variable corresponds to a node and the number obtained by the translation of the optimization variables is the index in the adjacency list for the first codification technique or in the depth-based list for the second. In this example, the adjacency-based codification of the network presented in Figure 5 is used, but it can be extended easily to the other method.

Figure 7 shows the decodification of two different sets of optimization variables: in the first one (Figure 7a), the resulting network is connected, while, in the second one (Figure 7b), it is not and, thus, the solution is unfeasible.

If the network is feasible, the calculation of the cost is run. The first activity performed is the estimation of the information packages through each node. This is computationally performed in the graph represented by the active connections by means of a customized search that starts from the deeper nodes and then goes upward. Each node creates one information package and forward the received ones. Figure 8 shows the transmission packages for each edge.



Optimization variables (a)

0,01	0,05	0,09	0,98	0,95	0,48	0,64	0,27	0,35	0,32

Adjacency list indexes (a)

1	1	1	4	5	2	3	2	2	2

Optimization variables (b)

0,12	0,02	0,43	0,34	0,98	0,97	0,89	0,24	0,96	0,32

Adjacency list indexes (b)

2	1	4	2	5	3	4	2	4	2

(**a**) Connected network. (**b**) Disconnected network.

Figure 7. Example of the decodification of the optimization variables: (**a**) feasible network; and (**b**) disconnected network.

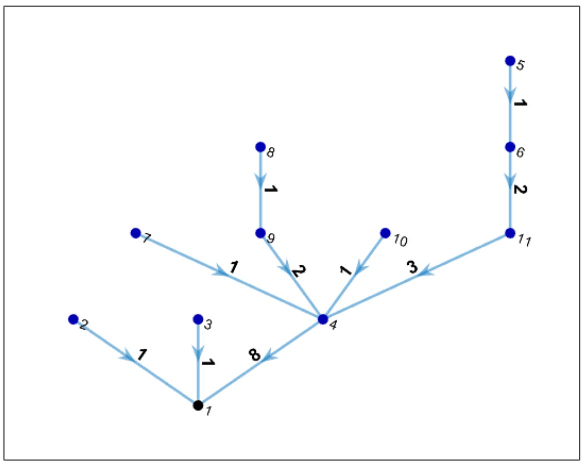


Figure 8. Representation of the active paths in the WSN with the number of transmitted packages for each edge.

Once the number of packages transmitted and received is computed, the energy model shown in Section 3.1 can be easily applied.

3.4. Optimization Environment

After having defined the WSN model and the codification technique, it is possible to design the optimization environment, i.e., all the interactions between the different elements involved in the solution of this problem.

The optimization environment designed for the routing problem is the one shown in Figure 9. Two evolutionary algorithms are used, PSO and SNO. The optimization variables used in these algorithms are decodified to a specific network configuration with the techniques shown in the previous section.

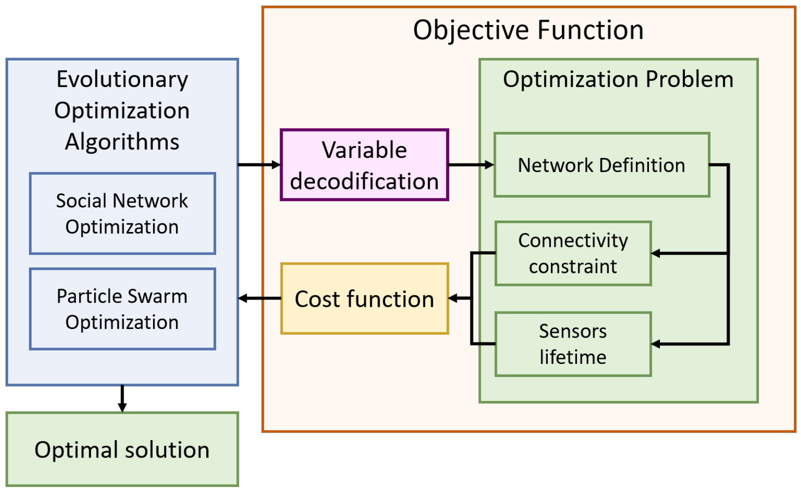


Figure 9. Optimization environment for the analyzed problem.

While a candidate solution of the algorithm is translated in a network configuration, two values are computed: the number of disconnected sensors, which is used as a constraint on the optimization problem, and the energy required by each sensor, which is the major objective of this problem.

The cost function is constructed for imposing the constraint and for helping the convergence of the algorithm.

In particular, it is defined as follows:

$$c = \begin{cases} 70 + 20 \cdot n_{disc} & n_{disc} > 0 \\ 10^5 \cdot \left(\max_i E_i + \dfrac{\sum_i E_i}{\lambda \cdot N} \right) & n_{disc} = 0 \end{cases} \tag{8}$$

The first condition is related to the compliancy with the constraints. The offset value (70) is used to avoid feasible solutions having higher costs than unfeasible ones: this creates a step in the convergence curves that helps the identification of the time in which the algorithm has reached the feasible region of the search space.

The second cost term is the one related to feasible solution. The scale factor 10^5 does not affect the optimization process; the cost term $\max_i E_i$ is the maximum energy consumption of the sensors and it is the real objective of the optimization problem, while the cost term $\sum_i E_i$ is the total energy consumed, and it is added, properly weighted by λ and the number of sensors N, for improving the convergence.

4. Results and Discussion

The optimization environment defined in the previous section has been applied for the routing optimization.

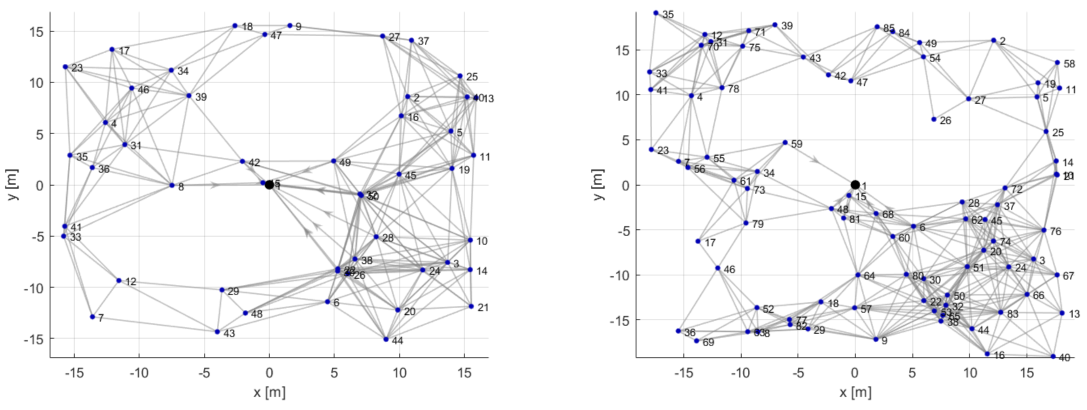
In this section, the results obtained are shown and discussed: in particular, the selected test cases are firstly analyzed; then, the optimization results of PSO and SNO are provided; and, finally, a peculiar case is analyzed in depth.

4.1. Test Cases

Due to the fact that the performance of the algorithms and codification methods can be highly case-dependant, several test cases are here created for the performance analysis.

Each network is characterized by a different number of sensors and a different deployment of them in the space, as shown in Figure 10, where a network with 50 nodes and one with 85 are compared.

The two networks are very different: in particular, the network with 85 nodes has a higher maximum node depth due to the upper area in which several nodes are not connected with the cluster head.



(**a**) 50 nodes network.

(**b**) 85 nodes network.

Figure 10. Examples of tested networks: network with: (**a**) 50 nodes; and (**b**) 85 nodes.

For each tested network, it is possible to calculate the maximum node depth, the total number of possible configurations with the adjacency-based codification, and the number of configurations with the depth-based codification.

Table 2 shows the parameters of the tested networks. The characteristics of each network does not depend just on the number of nodes: thus, the selected networks explore different possible combinations between node depth, number of connections, and number of nodes.

Analyzing the table, it is clear that the depth-based codification reduces drastically the size of the problem.

Table 2. Summary of all the tested networks.

Number	**Maximum**	**Number**	**Number of Adjacency-Based**	**Number of Depth-Based**
of nodes	node depth	of connections	configurations	configurations
ine 20	4	54	1.51×10^{13}	192
25	4	102	1.08×10^{21}	2.4×10^{6}
30	3	157	4.3×10^{28}	1.63×10^{7}
35	4	143	4.36×10^{29}	8.82×10^{9}
40	4	192	4.44×10^{35}	4.12×10^{10}
45	4	226	3.87×10^{42}	2.93×10^{12}
50	5	253	4.64×10^{47}	4.46×10^{15}
55	6	314	4.8×10^{54}	9.63×10^{18}
60	4	341	3×10^{60}	7.1×10^{20}
65	6	250	2.17×10^{53}	1.59×10^{12}
70	7	215	1.6×10^{52}	9.78×10^{13}
75	4	547	7.94×10^{84}	8.8×10^{25}
80	5	549	2×10^{87}	1.16×10^{28}
85	7	410	2.22×10^{79}	1.29×10^{27}
90	5	650	1.95×10^{101}	1.33×10^{38}
95	6	501	2.08×10^{92}	3.58×10^{32}
100	5	792	2.96×10^{116}	2.61×10^{46}
105	4	831	3.51×10^{122}	1.92×10^{46}
110	6	506	2.02×10^{102}	4.79×10^{31}
115	5	987	2.64×10^{137}	1.52×10^{52}
120	8	436	7.5×10^{98}	8.62×10^{28}
125	4	1129	3.7×10^{152}	9.16×10^{63}
130	5	1000	6.63×10^{149}	4.62×10^{62}
135	8	581	1.23×10^{120}	4.45×10^{36}
140	5	1423	1.68×10^{179}	3.93×10^{75}
145	5	1459	3.27×10^{184}	2.83×10^{77}
150	8	784	2.5×10^{147}	5.38×10^{52}

4.2. Results

To compare the results of the two different codification techniques and the two algorithm, the number of objective function calls was selected as termination criterion.

In fact, this parameter is the one that mainly drives the computational time in the optimization due to the fact that the self-time of the optimization algorithm can be usually neglected. For example, in the optimization of an 85-node network with 14,000 objective function calls, the self-time of SNO is 1.3 s for a total optimization time of 26 s, i.e., 5%. PSO is slightly faster and its self-time in the same conditions is 0.9 s, i.e., 3.5%.

Since the problem complexity grows with the number of sensors, the allowed number of objective function calls was also increased with the following empirical rule:

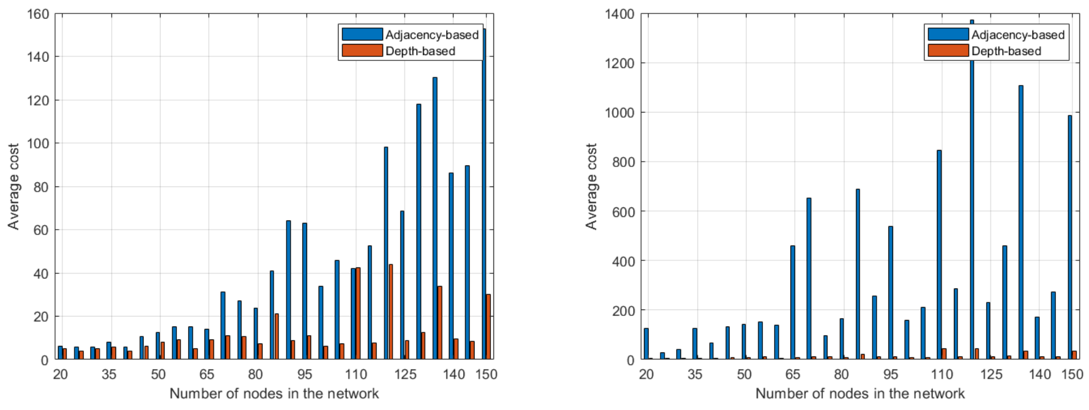
$$n_{call} = 1000 \cdot \left(\frac{N-20}{5} + 1 \right) \tag{9}$$

where N is the number of sensors in the network.

The population size for both these algorithms was selected according to a preliminary parametric analysis on some standard mathematical benchmarks. In particular, for PSO, the optimal population size is 50 individuals, while for SNO it is 100.

Due to the intrinsic stochastic nature of both these algorithms, 50 independent trials were performed for each configuration.

Figure 11 shows the results of the optimization. In particular, the average cost value obtained in the 50 independent trials is reported for each test case with the two codification techniques.



(**a**) SNO. (**b**) PSO.

Figure 11. Average cost value of 50 independent trials as function of the number of nodes in the network: (**a**) results of SNO; and (**b**) results of PSO.

Analyzing these results, it is possible to make some considerations. The effect of the problem codification technique impacts on the performance of the algorithms in different ways: in fact, for PSO, the depth-based codification is always much better than the adjacency-based one, while, for SNO, the superiority of the depth-based codification is less important. This is due to the fact that SNO usually performs better than PSO in handling high-dimensional problems with many local minima.

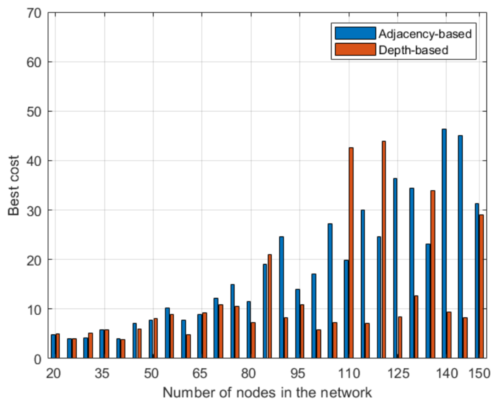
To introduce a numerical estimation of the difference between these two codification techniques, the gap value was calculated for each case:

$$\text{GAP} = \frac{c_{adj} - c_{depth}}{c_{adj}} \tag{10}$$

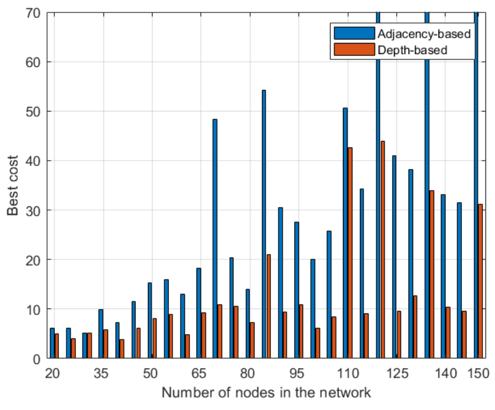
This value represents the improvement of the depth-based codification expressed as function of the adjacency-based value.

The average GAP value for the mean value of SNO over all the test cases is 0.58 meaning that the depth-based mean value is, on average, half of the adjacency-based one. For PSO, the average GAP value over the mean value is 0.98.

Figure 12 shows the best results obtained by both SNO and PSO. Analyzing this figure, two additional considerations can be done. The minimum values obtained by PSO are much lower than the average ones, indicating a difficulty of this algorithm in achieving the global minimum of the function. Secondly, the difference between the two codifications is reduced, in particular for the results of SNO: in fact, for this algorithm, in eight cases, the adjacency-based codification is better, and in only eight other cases the depth-based is drastically better. This can also be seen from the average GAP calculated on the minimum value, that is 0.18 for SNO. This means that the optimal solution requires some hops that violate the depth rule used to create the second codification.



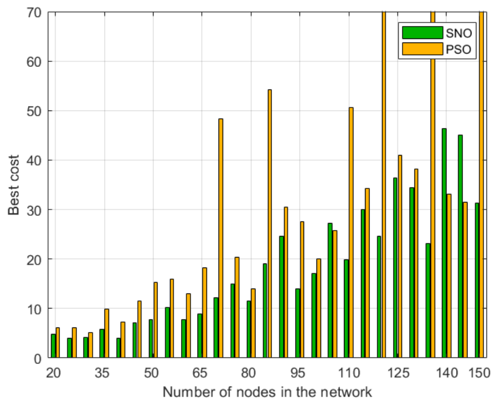
(**a**) SNO.



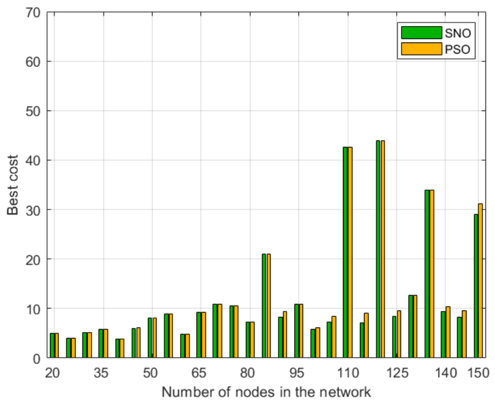
(**b**) PSO.

Figure 12. Best cost value of 50 independent trials as function of the number of nodes in the network: (**a**) results of SNO; and (**b**) results of PSO.

For comparing the results of the two algorithms, in particular, the best value obtained in the 50 independent trials, the results are plotted in Figure 13 underlining the comparison between the two algorithms.



(**a**) Adjacency-based codification.



(**b**) Depth-based codification.

Figure 13. Best cost value of 50 independent trials as function of the number of nodes in the network: (**a**) adjacency-based codification; and (**b**) depth-based codification.

Figure 13a shows that the performance of SNO with the adjacency-based codification is drastically better with respect to the ones of PSO, especially in nine cases in which the gap between the two algorithms overcomes 1; in only one case, PSO is better than SNO (gap -0.05). The average gap between the two algorithms is 2.95, highly biased by three cases in which it is over 10.

On the other hand, the results of Figure 13b show that with the depth-based codification the performance of the two algorithm becomes more similar. In fact, the average gap is 0.04 and in 18 cases the solutions obtained by the two algorithms have the same cost value.

To highlight some differences between the optimizers in this case, in which the results seem the same, the termination criterion has been modified in the simulations.

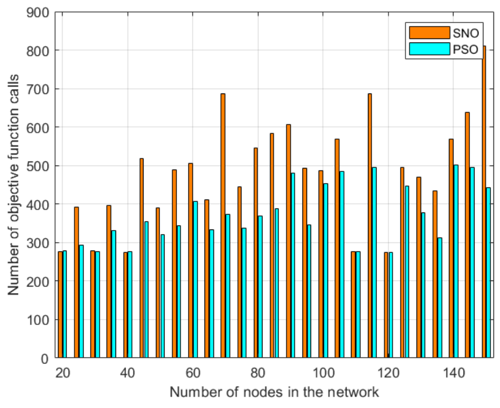
In single-objective optimization, the choice of the termination criterion is not as critical as in multi-objective optimization [34], however different possibilities are analyzed in the literature.

The most common criterion is the number of objective function calls: in this case, the computational time required by the optimization is limited a priori. A second possibility is related to the population diversity: when it drops, the algorithm has reached a minimum. The effectiveness of this criterion depends highly on the mutation operators of the algorithm and it is hardly applicable for SNO. Finally, another common possibility is to fix a maximum number of iterations in which the best solution is not improved [35]. In this last analysis, the non-improving iterations criterion was adopted.

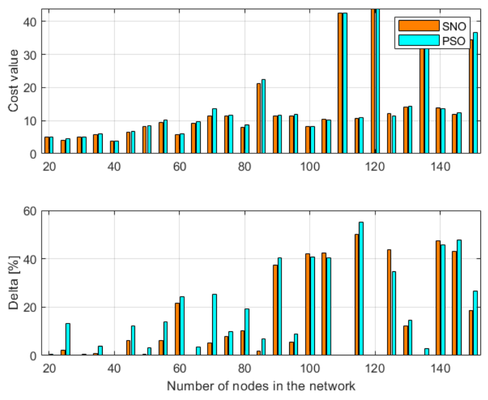
Figure 14 shows the results when the termination criterion was set to 10 non-improving iterations. In addition, in this case, 50 independent trials were performed for both algorithms.

The number of objective function calls required before the termination criterion was applied are shown in Figure 14a. It clearly shows that the effective number of objective function calls is much smaller to the time given in the standard optimization. It is interesting to see that PSO seems slightly faster than SNO.

The time performance should also be combined with the final cost values, as shown in Figure 14b, in which the final cost and the delta with respect to the previously found optimum are shown. These data show that the final cost of PSO is almost always greater than the one of SNO: this means that SNO performs more exploration than PSO. Finally, from the delta values, it is possible to notice that this second termination criterion often is not able to provide the algorithm enough time to find the optimum of the function.



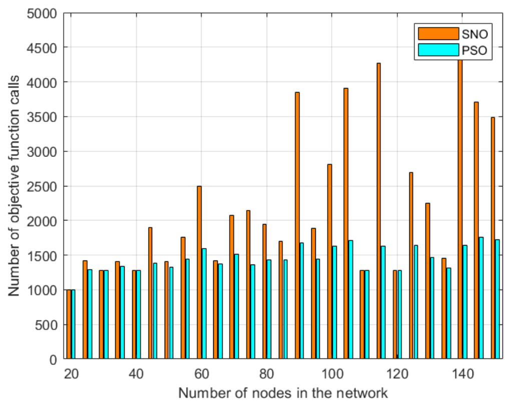
(**a**) Optimization time in objective function calls.



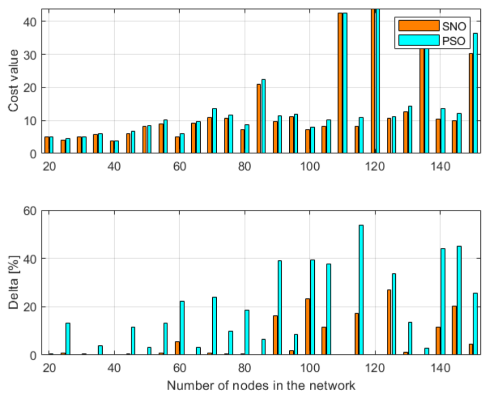
(**b**) Final cost and delta with respect to the standard optimization.

Figure 14. Optimization with 10 non-improvement iterations termination criterion: optimization time (**a**); and optimization cost (**b**).

To analyze the effect of the termination criterion, the number of non-improving iterations was increased to 50. The optimization time (Figure 15a) is more than doubled for both the algorithms. In addition, in these tests, in many cases, SNO requires more iterations than PSO. In Figure 15b, it is clear that, in almost all cases, SNO is able to achieve the optimal solutions, while the errors of PSO are still high.



(**a**) Optimization time in objective function calls.



(**b**) Final cost and delta with respect to the standard optimization.

Figure 15. Optimization with 50 non-improvement iterations termination criterion: optimization time (**a**); and optimization cost (**b**).

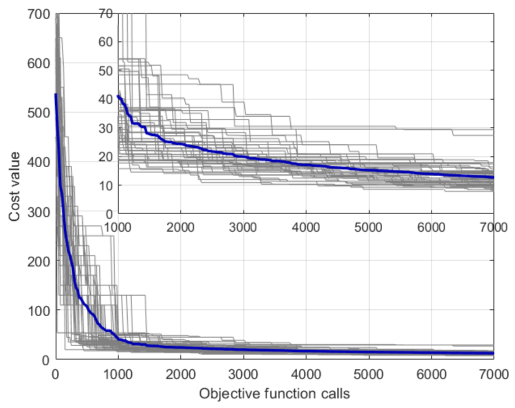
In the following, a peculiar network is deeply analyzed: its peculiarity is that the adjacency-based codification gives better results.

4.3. Analysis of a Peculiar Case

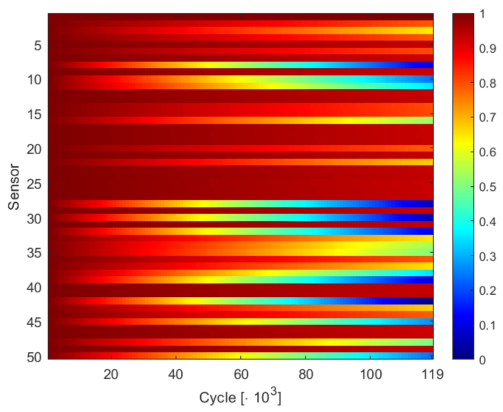
The network with 50 nodes is here analyzed as a peculiar case, representative of all the networks in which the adjacency-based codification gives better results with respect to the depth-based.

Here, the results of the four optimizations on this network (two codifications and two algorithms) are analyzed. All these results were obtained performing 50 independent trials and using as termination criterion 7000 objective function calls.

Figure 16a shows the convergence curves obtained with SNO. Each grey line is a single trial, while the blue line is the average convergence. The figure also provides a zoom of the last 6000 calls. The convergence curves show some jumps in the first iterations (before 1500 objective function calls), corresponding to the achievement of feasible solutions.



(**a**) Convergence curves of SNO



(**b**) Residual battery capacity for all the sensors

Figure 16. SNO optimization of the 50-nodes network with the adjacency-based codification: (**a**) convergence curves; and (**b**) residual battery capacity.

The performance of the best network achieved with this optimization is shown in Figure 16b, where the evolution of the sensors' battery capacity is represented. In this figure, the dark red color means full charge and blue empty battery. In this figure, it is possible to notice that the network lifetime is around 119 KCC and that six sensors are more critical than the others.

The active paths of the optimal solution are shown in Figure 17 with the green lines. In this figure, two areas have been highlighted. The red circles underline some sub-optimal configurations of the network: for example, in the left one, node 41 hops on 36 that hops on 33 that send all this information to sensor 35. This is sub-optimal because all these nodes could be connected directly to node 35.

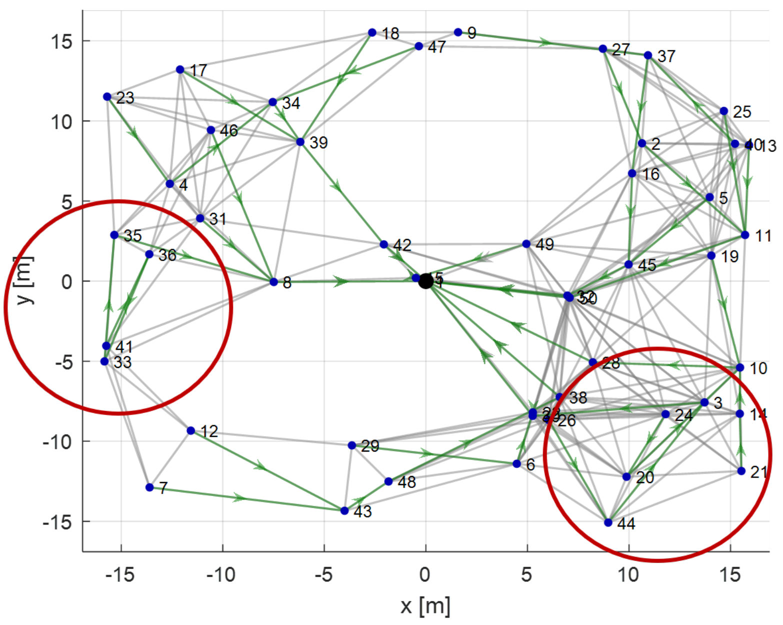
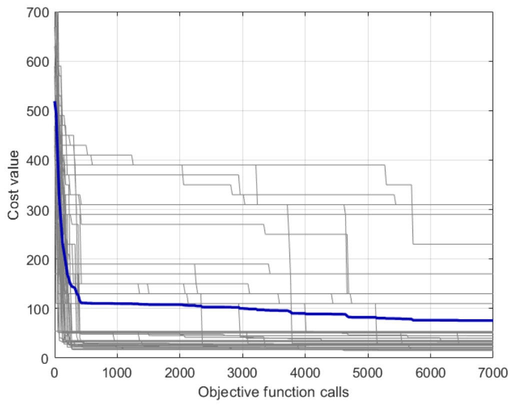


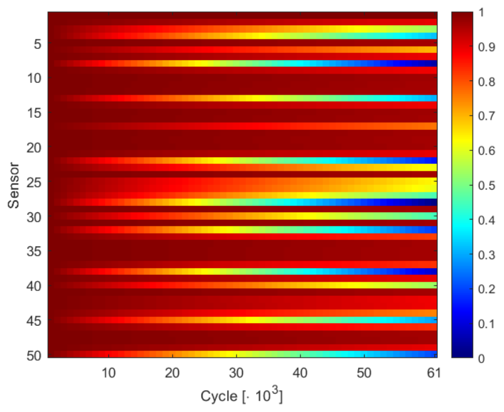
Figure 17. Routing in the optimal network. The green lines are the selected paths and the red circle underline some sub-optimal areas of the network.

The reason this sub-optimality appears is because all the involved nodes are not critical in the definition of the lifetime: in fact, as shown in Figure 16b, all of them achieve a minimum energy greater than the 50% of the initial capacity.

The convergence curves of PSO, represented in Figure 18a, shows that this algorithm could not find feasible solutions in all the independent trials. The stagnation in high-cost local minima makes the final optimal not competitive with the one found by SNO: in fact, as expressed by Figure 18b, the network lifetime is limited to 61 KCC. The reason of this very limited lifetime can be understood analyzing the active paths in the optimal solutions found by PSO shown in Figure 19. From the energy evolution it is possible to notice that the loaded nodes are the 28 and 8. In fact, they aggregates the information of large clusters of nodes: all the nodes highlighted in red transmit their information through node 8, while all the orange nodes hops on node 28.



(**a**) Convergence curves of PSO



(**b**) Residual battery capacity for all the sensors

Figure 18. PSO optimization of the 50-nodes network with the adjacency-based codification: (**a**) convergence curves; and (**b**) residual battery capacity.

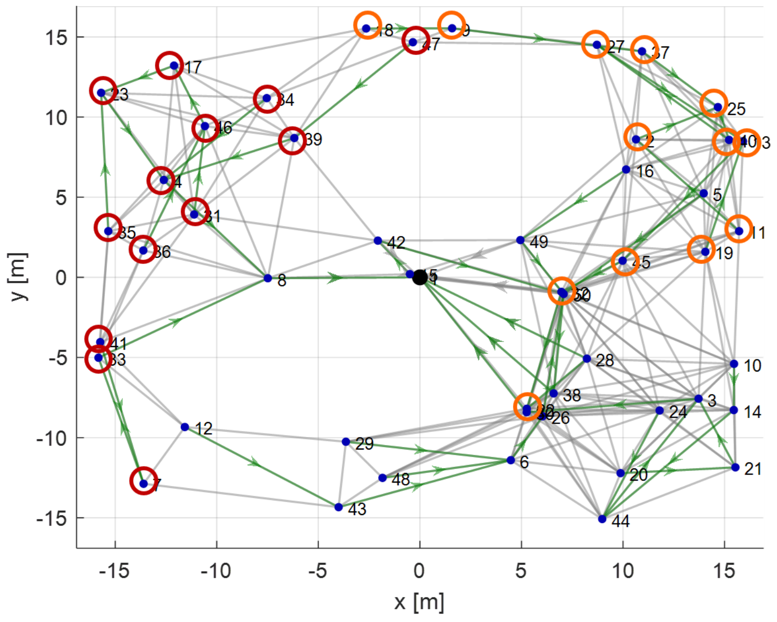
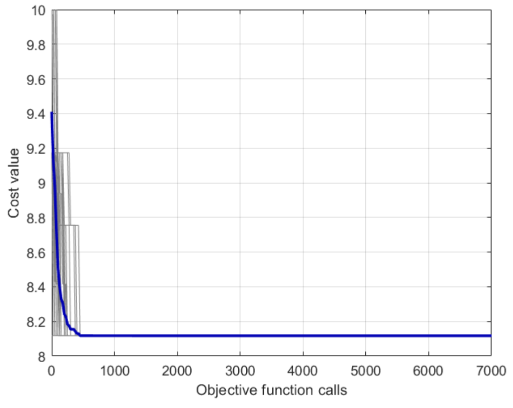
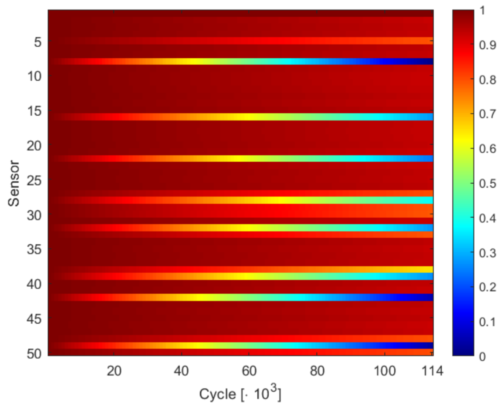


Figure 19. Routing in the optimal network. The green lines are the selected paths. The sensors with red circle all hop on node 8, and the orange circles correspond to nodes that hop on node 28.

The convergence of the 50 independent trials of SNO with the depth-based codification are shown in Figure 20a: the optimization process is very effective because all the trials converge to the same solution before 500 objective function calls out of the 7000 allowed. The fast convergence of all the trials to the same solution suggests that it is the global minimum for this function.



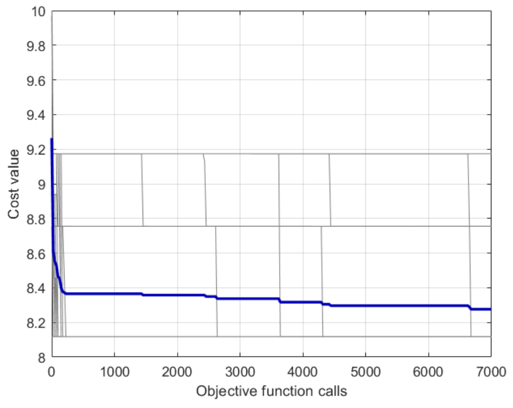
(**a**) Convergence curves of SNO



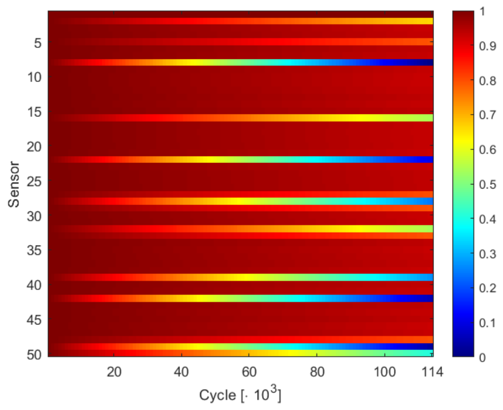
(**b**) Residual battery capacity for all the sensors

Figure 20. SNO optimization of the 50-nodes network with the depth-based codification: (**a**) convergence curves; and (**b**) residual battery capacity.

This assumption can be confirmed analyzing the convergence curves of the 50 trials of PSO (Figure 21a): in fact, most of the solutions converge toward the same minimum value. From these curves, it is possible to highlight that, even if the best solution of PSO is equal to the SNO one, the first algorithm is not able to converge in all the trials.



(**a**) Convergence curves of PSO



(**b**) Residual battery capacity for all the sensors

Figure 21. PSO optimization of the 50-nodes network with the depth-based codification: (**a**) convergence curves; and (**b**) residual battery capacity.

From the energy evolution of SNO (Figure 20b) and PSO (Figure 21b), it is possible to see that the network lifetime (114 KCC) is lower than the best solution achieved by SNO in the adjacency-based codification (119 KCC). In fact, in these solutions, the transmission load is concentrated mostly just in three nodes: nodes 8, 42, and 49.

The overload of these nodes can bee seen from the optimal solution obtained with the depth-based codification shown in Figure 22. The colored circles underline the cluster of nodes that transmits through the highly loaded nodes. In particular, the red circles belong to the cluster of node 8, the yellow ones to sensor 42, and the orange ones to node 49.

This solution is the global minimum with the depth-based codification because the search space has been drastically reduced and, thus, the algorithm has no degree of freedom to reduce the load of critical nodes.

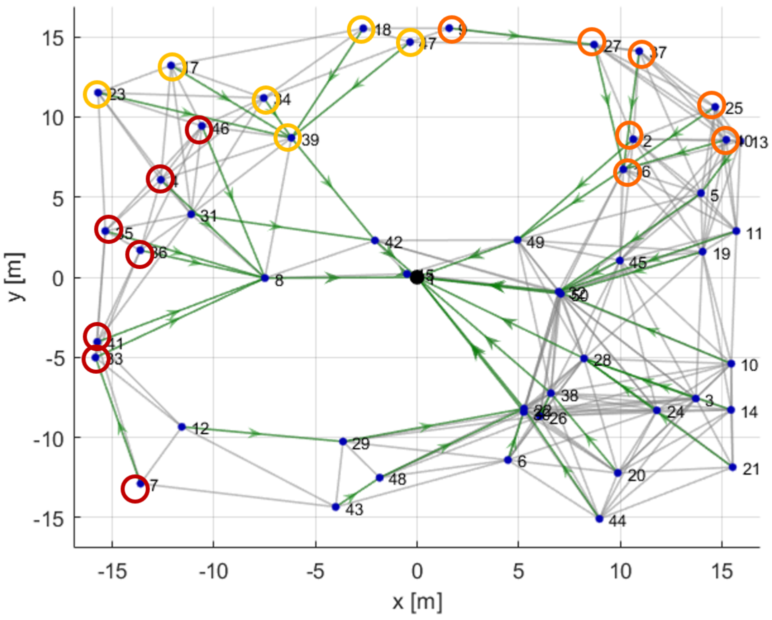


Figure 22. Routing in the optimal network with the depth-based codification. The sensors with red circle all hop on node 8, the orange circles correspond to nodes that hop on node 49, and the yellow ones hop on node 42.

5. Conclusions

In this paper, the maximization of the network lifetime of a Wireless Sensor Network is achieved by properly selecting the routing paths. This problem is faced with two different evolutionary optimization algorithms: the traditional PSO and the more recent SNO.

The objective of this paper is to analyze the difference between two different problem codification methods: with the first one (the adjacency-based codification), the algorithm can choose an "exit way" among all the adjacent nodes, while, in the second one (the depth-based codification), the selection is restricted to sensors with lower depth in the network.

Two important findings were obtained with the test campaign conducted over more than 25 test cases. Firstly, the reduction of the search space size due to the depth-based codification improves drastically the optimization convergence, in terms of both the quality of the final solution and the optimization time. This improvement is more evident in the PSO algorithm: in fact, with the adjacency-based codification, the performance of this algorithm is very poor, while, with the depth-based codification, it is comparable with SNO.

The drawback of the depth-based codification is due to the lower degrees of freedom that are given to the optimizer: in fact, while this reduces the problem complexity, it is likely to eliminate some good network configurations from the search space. This phenomenon was deeply inspected and it was confirmed by the results of the optimization of the 50-nodes network.

Author Contributions: Conceptualization, A.N., F.G., and M.M.; Data curation, A.N. and R.Z.; Formal analysis, A.G. and R.Z.; Funding acquisition, F.G. and A.G.; Investigation, M.M.; Methodology, A.N. and M.M.; Project administration, R.Z.; Resources, R.Z.; Software, A.N. and M.M.; Supervision, F.G. and A.G.; Validation, A.N. and F.G.; Visualization, A.N.; and Writing—original draft, A.N., F.G., M.M., A.G., and R.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

WSN	Wireless Sensor Network
KCC	Kilo Clock Cycles
SNO	Social Network Optimization
PSO	Particle Swarm Optimization

References

1. Buratti, C.; Conti, A.; Dardari, D.; Verdone, R. An overview on wireless sensor networks technology and evolution. *Sensors* **2009**, *9*, 6869–6896. [CrossRef] [PubMed]
2. Iacca, G.; Neri, F.; Caraffini, F.; Suganthan, P.N. A differential evolution framework with ensemble of parameters and strategies and pool of local search algorithms. In *European Conference on the Applications of Evolutionary Computation*; Springer: Berlin, Germany, 2014; pp. 615–626.
3. Grimaccia, F.; Gruosso, G.; Mussetta, M.; Niccolai, A.; Zich, R.E. Design of tubular permanent magnet generators for vehicle energy harvesting by means of social network optimization. *IEEE Trans. Ind. Electron.* **2017**, *65*, 1884–1892. [CrossRef]
4. Hassan, R.; Cohanim, B.; De Weck, O.; Venter, G. 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, TX, USA, 18–21 April 2005; p. 1897.
5. Baioletti, M.; Milani, A.; Poggioni, V.; Rossi, F. An ACO approach to planning. In *European Conference on Evolutionary Computation in Combinatorial Optimization*; Springer: Berlin, Germany, 2009; pp. 73–84.
6. Caraffini, F.; Neri, F.; Poikolainen, I. Micro-differential evolution with extra moves along the axes. In Proceedings of the 2013 IEEE Symposium on Differential Evolution (SDE), Singapore, Singapore, 16–19 April 2013; pp. 46–53.
7. Caraffini, F.; Kononova, A.V.; Corne, D. Infeasibility and structural bias in differential evolution. *Inf. Sci.* **2019**, *496*, 161–179. [CrossRef]
8. Piotrowski, A.P. Review of differential evolution population size. *Swarm Evol. Comput.* **2017**, *32*, 1–24. [CrossRef]
9. Baioletti, M.; Milani, A.; Santucci, V. Automatic algebraic evolutionary algorithms. In *Italian Workshop on Artificial Life and Evolutionary Computation*; Springer: Berlin, Germany, 2017; pp. 271–283.
10. Baioletti, M.; Bari, G.D.; Milani, A.; Poggioni, V. Differential Evolution for Neural Networks Optimization. *Mathematics* **2020**, *8*, 69. [CrossRef]
11. Simon, D. Biogeography-based optimization. *IEEE Trans. Evol. Comput.* **2008**, *12*, 702–713. [CrossRef]
12. Sandeep, D.; Kumar, V. Review on clustering, coverage and connectivity in underwater wireless sensor networks: A communication techniques perspective. *IEEE Access* **2017**, *5*, 11176–11199. [CrossRef]
13. Kuila, P.; Jana, P.K. A novel differential evolution based clustering algorithm for wireless sensor networks. *Appl. Soft Comput.* **2014**, *25*, 414–425. [CrossRef]
14. Zahedi, Z.M.; Akbari, R.; Shokouhifar, M.; Safaei, F.; Jalali, A. Swarm intelligence based fuzzy routing protocol for clustered wireless sensor networks. *Expert Syst. Appl.* **2016**, *55*, 313–328. [CrossRef]
15. Chen, Y.; Li, D.; Ma, P. Implementation of Multi-objective Evolutionary Algorithm for Task Scheduling in Heterogeneous Distributed Systems. *JSW* **2012**, *7*, 1367–1374. [CrossRef]
16. Page, A.J.; Keane, T.M.; Naughton, T.J. Multi-heuristic dynamic task allocation using genetic algorithms in a heterogeneous distributed system. *J. Parallel Distrib. Comput.* **2010**, *70*, 758–766. [CrossRef] [PubMed]
17. Ferjani, A.A.; Liouane, N.; Kacem, I. Task allocation for wireless sensor network using logic gate-based evolutionary algorithm. In Proceedings of the 2016 International Conference on Control, Decision and Information Technologies (CoDIT), St. Julian's, Malta, 6–8 April 2016; pp. 654–658.
18. Niccolai, A.; Grimaccia, F.; Mussetta, M.; Zich, R. Optimal Task Allocation in Wireless Sensor Networks by Means of Social Network Optimization. *Mathematics* **2019**, *7*, 315. [CrossRef]
19. Ho, J.H.; Shih, H.C.; Liao, B.Y.; Chu, S.C. A ladder diffusion algorithm using ant colony optimization for wireless sensor networks. *Inf. Sci.* **2012**, *192*, 204–212. [CrossRef]

20. Zhang, H.; Li, Z.; Shu, W.; Chou, J. Ant colony optimization algorithm based on mobile sink data collection in industrial wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.* **2019**, *2019*, 152. [CrossRef]
21. Hu, X.M.; Zhang, J.; Yu, Y.; Chung, H.S.H.; Li, Y.L.; Shi, Y.H.; Luo, X.N. Hybrid genetic algorithm using a forward encoding scheme for lifetime maximization of wireless sensor networks. *IEEE Trans. Evol. Comput.* **2010**, *14*, 766–781. [CrossRef]
22. Caputo, D.; Grimaccia, F.; Mussetta, M.; Zich, R.E. An enhanced GSO technique for wireless sensor networks optimization. In Proceedings of the IEEE Congress on Evolutionary Computation, Hong Kong, China, 1–6 June 2008; pp. 4074–4079.
23. Caputo, D.; Grimaccia, F.; Mussetta, M.; Zich, R.E. Genetical swarm optimization of multihop routes in wireless sensor networks. *Appl. Comput. Intell. Soft Comput.* **2010**, *2010*, 523943. [CrossRef]
24. Omidvar, A.; Mohammadi, K. Particle swarm optimization in intelligent routing of delay-tolerant network routing. *EURASIP J. Wirel. Commun. Netw.* **2014**, *2014*, 147. [CrossRef]
25. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell.* **2007**, *1*, 33–57. [CrossRef]
26. Simon, D. *Evolutionary Optimization Algorithms*; John Wiley & Sons: Hoboken, NJ, USA, 2013.
27. Engelbrecht, A. Particle swarm optimization: Velocity initialization. In Proceedings of the 2012 IEEE Congress on Evolutionary Computation, Brisbane, QLD, Australia, 10–15 June 2012; pp. 1–8.
28. Trelea, I.C. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Inf. Process. Lett.* **2003**, *85*, 317–325. [CrossRef]
29. Niccolai, A.; Grimaccia, F.; Mussetta, M.; Pirinoli, P.; Bui, V.H.; Zich, R.E. Social network optimization for microwave circuits design. *Prog. Electromagn. Res.* **2015**, *58*, 51–60. [CrossRef]
30. Niccolai, A.; Grimaccia, F.; Mussetta, M.; Zich, R. Modelling of interaction in swarm intelligence focused on particle swarm optimization and social networks optimization. *Swarm Intell.* **2018**, pp. 551 - 582.
31. Grimaccia, F.; Mussetta, M.; Niccolai, A.; Zich, R.E. Optimal computational distribution of social network optimization in wireless sensor networks. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–7.
32. Park, J.; Sahni, S. Maximum lifetime routing in wireless sensor networks. *IEEE/ACM Trans. Netw.* **2004**, *12*, 609–619.
33. Dong, Q. Maximizing system lifetime in wireless sensor networks. In Proceedings of the IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, Boise, ID, USA, 15 April 2005; pp. 13–19.
34. Wong, J.Y.; Sharma, S.; Rangaiah, G. Design of shell-and-tube heat exchangers for multiple objectives using elitist non-dominated sorting genetic algorithm with termination criteria. *Appl. Therm. Eng.* **2016**, *93*, 888–899. [CrossRef]
35. Eiben, A.E.; Smith, J.E. *Introduction to Evolutionary Computing*; Springer: Berlin, Germany, 2003; Volume 53,.