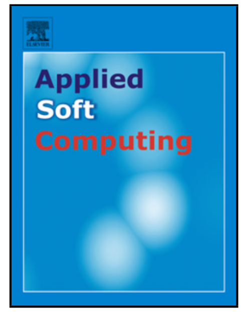
Accepted Manuscript



Title: Static and Dynamic Minimum Energy Broadcast Problem in Wireless Ad-hoc Networks: A PSO-based Approach and Analysis

Author: Ping-Che Hsiao Tsung-Che Chiang Li-Chen Fu

PII:	S1568-4946(13)00274-3
DOI:	http://dx.doi.org/doi:10.1016/j.asoc.2013.08.008
Reference:	ASOC 2038
To appear in:	*Applied Soft Computing*
Received date:	2-10-2012
Revised date:	4-7-2013
Accepted date:	10-8-2013

Please cite this article as: P.-C. Hsiao, T.-C. Chiang, L.-C. Fu, Static and Dynamic Minimum Energy Broadcast Problem in Wireless Ad-hoc Networks: A PSO-based Approach and Analysis, *Applied Soft Computing Journal* (2013), http://dx.doi.org/10.1016/j.asoc.2013.08.008

Static and Dynamic Minimum Energy Broadcast Problem in Wireless Ad-hoc Networks: A PSO-based Approach and Analysis

Ping-Che Hsiao[a], Tsung-Che Chiang[b], and Li-Chen Fu[ac*]

[a]Department of Computer Science and Information Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei 10617 Taiwan, R.O.C.

[b]Department of Computer Science and Information Engineering, National Taiwan Normal University, No.88, Sec. 4, Tingzhou Rd., Wenshan District, Taipei 116, Taiwan, R.O.C.

[c]Department of Electrical Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei 10617 Taiwan, R.O.C.

Abstract. In this paper, we address the minimum energy broadcast (MEB) problem in wireless ad-hoc networks (WANETs). The researches in WANETs have attracted significant attentions, and one of the most critical issues in WSNs is minimization of energy consumption. In WANETs the packets have to be transported from a given source node to all other nodes in the network, and the objective of the MEB problem is to minimize the total transmission power consumption. A hybrid algorithm based on particle swarm optimization (PSO) and local search is presented to solve the MEB problem. A power degree encoding is proposed to reflect the extent of transmission power level and is used to define the particle position in PSO. We also analyze a well-known local search mechanism, *r*-shrink, and propose an improved version, the intensified *r*-shrink. In order to solve the dynamic MEB problem with node removal/insertion, this paper provides an effective simple heuristic, Conditional Incremental Power (CIP), to reconstruct the broadcast network efficiently. The promising results indicate the potential of the proposed methods for practical use.

Keywords: Particle Swarm Optimization; Minimum Energy Broadcast Problem; Wireless Ad-Hoc Network; Wireless Sensor Network; Dynamic Minimum Energy Broadcast Problem

Static and Dynamic Minimum Energy Broadcast Problem in Wireless Ad-hoc Networks: A PSO-based Approach and Analysis

Abstract. In this paper, we address the minimum energy broadcast (MEB) problem in wireless ad-hoc networks (WANETs). The researches in WANETs have attracted significant attentions, and one of the most critical issues in WSNs is minimization of energy consumption. In WANETs the packets have to be transported from a given source node to all other nodes in the network, and the objective of the MEB problem is to minimize the total transmission power consumption. A hybrid algorithm based on particle swarm optimization (PSO) and local search is presented to solve the MEB problem. A power degree encoding is proposed to reflect the extent of transmission power level and is used to define the particle position in PSO. We also analyze a well-known local search mechanism, *r*-shrink, and propose an improved version, the intensified *r*-shrink. In order to solve the dynamic MEB problem with node removal/insertion, this paper provides an effective simple heuristic, Conditional Incremental Power (CIP), to reconstruct the broadcast network efficiently. The promising results indicate the potential of the proposed methods for practical use.

Keywords: Particle Swarm Optimization; Minimum Energy Broadcast Problem; Wireless Ad-Hoc Network; Wireless Sensor Network; Dynamic Minimum Energy Broadcast Problem

1. Introduction

A wireless ad-hoc network (WANET) is a collection of nodes dynamically forming a connected network without relying on any preexisting infrastructure. There has been an increasing interest in the use of WANETs due to their convenient deployment, flexible communication, and a variety of applications. Wireless sensor network (WSN), which is composed of a large number of sensor nodes, is one of the typical manners of WANET. These nodes are ordinarily equipped with sensing, communication, and computing abilities. Each sensor node can measure environmental parameters such as temperature, humidity, sound, and vibration, perform simple computations, and communicate with neighboring nodes or base station. WSN have already been applied broadly on civil and military applications including forest surveillance, factory automation, disaster monitoring, border protection, battle field surveillance, and animal tracking [1-4].

Generally speaking, each sensor node in the network has limited energy (usually a battery or an embedded form of energy resource), which is in some cases completely non-rechargeable or non-renewable [5]. It connotes that these nodes are likely to be on their tasks for a long time without reorganization or provision. One of the utmost issues in WANETs is the determination of network routing. In the routing problems, we have to maximize the network lifetime; in the meantime, we have to prevent the loss of network connectivity. Compared with traditional link-based networks, WANETs can be deployed costlessly in a very short time, and they do not depend on the preexistent facilities, e.g., basement station or router. Different from traditional networks, in WANETs there is usually more than one receiver in a single transmission. The packets are receivable for all the nodes which locate in the transmission range of the sender node. This evident difference is also called Wireless Multicast Advantage (WMA) [6]. Hence, the network construction algorithms used in the link-based networks cannot be forthright applied to WANETs due to their differences in innate transmission properties.

The minimum energy broadcast (MEB) problem is one of the important scenarios in WANETs, where the packets need to be disseminated from the source node to all other nodes. The MEB problem is aimed at minimizing the total energy consumption, and it has been proven to be NP-complete [7], [8]. This paper intends to provide a solution to the MEB problem in WANETs. We take advantage of the fast convergence nature of particle swarm optimization (PSO) to solve the problem. We propose the power degree to define the particle position. We then go a step further to analyze one well-known local search mechanism, *r*-shrink, and propose an improved version. Apart from the static scenario, we also study on the dynamic MEB problem where a number of nodes are added to/deleted from the network. We propose a simple heuristic, Conditional Incremental Power (CIP), to deal with the changing environment. Part of this study has been presented earlier in [9].

This paper is divided into six sections. Section 2 provides a brief introduction to the MEB problem and some existing literatures. We also review two present encoding mechanisms for the wireless network topology. Section 3 details the algorithm, and we present a repairing scheme to solve the dynamic MEB problem in Section 4. Section 5 presents the experimental results for benchmark instances. Finally, conclusions and some plans for future development are given in Section 6.

2. The minimum energy broadcast (MEB) problem

2.1. Problem description

In the network model with wireless fashion, the power required to send packets is

$$P = \xi \times d^{\alpha}, \tag{1}$$

where ξ is the power threshold, and d is the distance between the sender node and the receiver node. The variable α is the path loss exponent, depending on the transmission medium in the environment. It is usually set to a real number ranging between 2 and 4 [10]. All of the nodes located within the distance d from the sender node can receive the packets. Without loss of generality, power threshold ξ is normalized to 1, and the power required for sending packets from node i to node j can be reduced to

$$P_{ij} = d_{ij}^{\alpha}, \tag{2}$$

where d_{ij} is the distance between node i and node j. Then, the total power consumption $f(T)$ of a broadcast tree $T = (V, E)$ is defined by

$$f(T) = \sum_{i \in V} \max_{\{j | e_{ij} \in E\}} d_{ij}^{\alpha}, \tag{3}$$

where V is the set of sensor nodes in T, E is the set of directed edges in T, e_{ij} is the directed edge from node i to node j, standing for node j can receive the packets transmitted from node i. The broadcast tree represents routing paths from a specified source node s to all other nodes in V. The objective of this problem is to minimize the total power consumption in (3).

Before going any further, we would like to define the key terminological terms we will be using in this paper.

Definition 1. (critical child). The transmission power level of a sensor node hinges on the *critical child*, which is the farthest child node of this sensor node [11]. A critical child $cr(i)$ of node i can be represented by

$$cr(i) = \arg \max_{\{j | e_{ij} \in E\}} d_{ij}^{\alpha}, \tag{4}$$

Definition 2. (leaf). A broadcast tree must contain a number of *leaf* nodes, which do not need to retransmit the packets to other nodes. They don't need any power consumption for transmission.

Definition 3. (ascendant and descendant). In a broadcast tree, if node i is on the path from the source node s to node j, then node i is an ascendant node of node j; on the contrary, node j is a descendant node of node i [12]. The source node s, therefore, is the ascendant node of all other nodes in the network.

Sometimes, the energy consumption in transmission can be reduced by using one or more intermediate nodes to indirectly transmit packets [13]. Nonetheless, a far-reaching transmission is sometimes preferred in the broadcast scenario. The more spacious the transmission range is, the more nodes can receive the packets. For instance, in Fig. 1(a) node *A* broadcasts through multiple intermediate transmissions; on the contrary, in Fig. 1(b), node *A* uses a single long-distance transmission range to reach all nodes. The traditional link-based routing algorithms are not suitable for the MEB problem because they only consider uni-path routing, i.e., a single receiver in a transmission.

Fig. 1 << Insert Fig. 1 about here >>

2.2. Related Work

Various groups of researchers have worked with the MEB problem. The algorithms can be categorized into (a) simple heuristics, (b) local search algorithms, and (c) metaheuristics.

Although there is a substantial space for improvement in simple heuristics, they are popular since they are able to construct a broadcast solution in a very short time. Various simple constructive heuristics for the MEB problem have been proposed, and most of these works are based on Prim's algorithm [14-16], which is a greedy algorithm for finding a minimum weighted spanning tree. Some examples are Adaptive Broadcast Consumption (ABC) [17], Broadcast Incremental Power (BIP) [14], Greedy Perimeter Broadcast Efficiency (GPBE) [15], and Shortest Path Tree (SPT) [14]. These heuristics start by the source node and then repeat adding the links by a specific greedy function. For instance, BIP adds into the broadcast tree the node which brings the minimum incremental power. Some of previous studies claimed that their work is a heuristic, but they are actually combinations of existing simple heuristics and local search algorithms [18-20].

Recent research has also suggested that the use of local search can achieve more desirable solutions. Examples of notable local search methods include Sweep [14], Embedded Wireless Multicast Advantage (EWMA) [7], *r*-shrink [21], Broadcast Incremental-Decremental Power (BIDP) [22], and Largest Expanding Sweep Search (LESS) [23]. Among these algorithms, *r*-shrink, which reassigns the farthest *r* children to another node which produces the minimum incremental power, is the most widely used owing to its superior performance.

Metaheuristics have attracted significant attentions due to its abilities in solving large-scale problems. Many works have been done in several application areas in the WANETs [24-26]. Some works based on metaheuristics have been proposed to deal with the MEB problem, including Iterated Local Search Algorithm (ILS) [12], Evolutionary Local Search Algorithm (ELS) [27], Hybrid Genetic Algorithm (HGA) [28], and Ant Colony Optimization (ACO) [29-33]. In order to utilize metaheuristics algorithms, each solution has to be transformed into an encoded sequence in the search space. Common encoding schemes in WANET include power level encoding and permutation encoding.

2.2.1. Power level encoding

Some existing works in the research of WANETs use the power level encoding. It encodes a solution into a sequence of real values, which are transmission power levels of nodes [10]. A solution can be represented as

$$\left[P_1, P_2, P_3, ..., P_n\right], \tag{5}$$

where n is the number of nodes in the network and P_i is the transmission power level of node i.

Nevertheless, power level encoding has an undesirable feature. Fig. 2 shows an example of redundant solutions in the power level encoding. In Fig. 2(a), the critical child of node A is node B, but the transmission power level is greater than the required transmission power level; node C has the same problem, the transmission power level is greater than the required power to cover the critical child node D. Therefore, the power level encoding has to proceed with a reduction procedure after decoding. The broadcast tree after reduction can be seen in Fig. 2(b), where the transmission power level of each node is reduced to the precise value required to cover the critical child. Due to the above circumstance, there are a large number of redundant solutions which lead to the same broadcast tree. This kind of encoding will result in a large area of flat plateaus in the search space such that the search algorithm is hard to find good solutions.

Fig. 2 << Insert Fig. 2 about here >>

2.2.2. Permutation encoding

Permutation encoding [34] is also one common way for representing a broadcast tree. In the permutation encoding, a sequence of node ID comprises a solution:

$$\boldsymbol{\pi} = [\pi_1, \pi_2, \pi_3, \ldots, \pi_n], \tag{6}$$

where $\boldsymbol{\pi}$ is a permutation of IDs 1, 2, 3, …, n. The basic concept of the transformation is using BIP [14] to link the nodes in the order they appear in the permutation. The permutation encoding is also used in several constrained minimum spanning tree problems.

The search space of the permutation encoding does not contain all feasible solutions. By using BIP to link nodes, many broadcast trees cannot be constructed by any permutation. It hasn't been proven that the optimal solution is included in the search space of permutation encoding. Furthermore, it is difficult to conversely transform a broadcast tree into a permutation sequence. Therefore, Lamarckian learning is not possible.

3. PSO-based approach for the static MEB problem

Our proposed approach is based on PSO and an improved version of *r*-shrink. This section firstly introduces the proposed power degree encoding scheme. Then, we detail the steps in PSO, including the proposed intensified r-shrink procedure. The entire approach is summarized in the last subsection.

3.1. Particle swarm optimization

Particle swarm optimization (PSO) was originally proposed by Kennedy and Eberhart [35]. In PSO, each solution is described as the position of a *particle* in the search space, and the particle is intended to find the optimal solution by simulating the movement of a bird flock or fish school. These particles flying around in the search space aim to find the global optimum. It is assumed that particles are sharing information between one another. The movement of a particle is decided by the *velocity*, which guides the particle toward the best-known positions. The velocity of particle i can be calculated by

$$v_i' = wv_i + r_1c_1(x_i^{pb} - x_i) + r_2c_2(x^{gb} - x_i), \tag{7}$$

and the new position of ith particle can be then updated by v_i':

$$x_i' = x_i + v_i', \tag{8}$$

where v_i' and v_i are the new and original velocity vectors, respectively. The variable x_i^{pb} represents the individual best-known position of particle i, also called *pbest*; x^{gb} represents the global best-known position, also called *gbest*; w is an inertia weight; c_1 and c_2 are weights for acceleration towards *pbest* and *gbest*; x_i' and x_i are the new and original positions of particle i, respectively; r_1 and r_2 are random numbers between 0 and 1.

3.2. Power degree encoding

Here, we present a new encoding scheme called *power degree* encoding. We define *power degree* as the extent of the transmission power level of a node. Power degree and power level are intuitively alike. Nonetheless, instead of using the real value of the transmission power level, the power degree uses discrete integer to deal with overmuch redundant solutions. Furthermore, it can transform the solution between a broadcast tree and an encoding sequence.

Definition 4. (power degree). For each node in the network, we sort the neighboring nodes except the source node in the non-decreasing order of distance. Then, the power degree is the position of the critical child in the sorted sequence. If a node is a leaf node, then its power degree is zero.

Let us illustrate the idea by Fig. 3. The sorted sequence of neighboring nodes from near to distant with respect to the source node S is $\{D, A, C, B\}$. Consequently, if the critical child of S is D, the power degree is 1. If the critical child is A, C, or B, the power degree is 2, 3, or 4, respectively. The power degree of the remaining nodes can be determined in the same way. After the power degree table is set up, we can then encode a broadcast tree into a power degree sequence. The power degree is decided by each node's critical child, and the encoding is the sequence of power degrees. Given a broadcast tree such as Fig. 4, the power degree encoding with five nodes is $\{3, 0, 0, 1, 0\}$. The fitness of a solution is computed by (3), the power consumption of the network.

Fig. 3 << Insert Fig. 3 about here >>

Fig. 4 << Insert Fig. 4 about here >>

3.3. Acceleration

In our algorithm, we update the velocity by the following formula:

$$v_i' = wv_i + r_1(x_i^{pb} - x_i) + r_2(x_i^{nb} - x_i), \tag{9}$$

where x_i^{nb} represents the neighborhood-best solution, also called *nbest*, which is the best among the x_j^{pb} of the K nearest particles j from particle i. Several literatures indicated that using the global best solution (x^{gb}) leads to high search speed but meanwhile also tends to cause premature convergence. This is also observed in our preliminary experiments. Therefore, we use the neighborhood-best solution to replace the role of the global best solution. To find the nearest particles, the distance $d(x_i, x_j)$ between two particles x_i and x_j is calculated by the Euclidean distance of their power degree encoding vectors in (10):

$$d(x_i, x_j) = \sum_{k=1}^{n} \sqrt{\left(x_{ik} - x_{jk}\right)^2}, \tag{10}$$

where n is the number of nodes and x_{ik} represents the power degree of the kth node of the ith particle.

The coefficients c_1 and c_2 in the velocity update equation (7) in PSO control the weights of the velocity toward the local and global/neighborhood best solutions. After examining several possible combinations of values for them, we found that they only slightly affect the performance of our approach. To make our approach simple, we set them by 1 and save the two parameters. We confine the values of velocity vector to the range $[-v_{max}, v_{max}]$, which can prevent particles from incurring over-perturbation.

After updating the velocity vector, a mutation-like procedure *craziness* is going to assign random numbers to velocity vectors. Craziness [35] was used as a simulation of the lifelike interesting movement in a bird flock or other animal groups. It helps to avoid being trapped in the local optima. Each particle will undergo the craziness procedure in a probability p_c. If a particle undergoes the procedure, one element in its velocity vector is randomly chosen and set to a random value in $[-v_{max}, v_{max}]$.

3.4. Landing

After acceleration and craziness, we can then obtain the new position of each particle by (8). We group the results into three cases: (a) the power degree is the same as before and nothing needs to be done, (b) the power degree increases, and (c) the power degree decreases. In order to keep the solution feasible, the network topology has to be adjusted carefully.

Assume the power degree of one node is being increased, i.e. the transmission range is extended. This node will become the new parent of some nodes that are located in the extended region of the increased power. Fig. 5 is an example of increasing the power degree of node B from zero to one. Node B becomes the new parent of C in Fig. 5(b), and B is disconnected from S. To reconnect B, in Fig. 5(c) we choose S, which is the node needs the least additional power to transmit to B, to be the parent of B in the new broadcast tree. When we increase the power degree of a node i from d_{old} to d_{new}, we set the parent of the affected nodes by

$$pa(v) \leftarrow i, \quad \forall v \in \left\{ cr^d(i) \middle| \begin{array}{c} d_{old} < d < d_{new}, cr^d(i) \notin asendant(i) \\ d = d_{new} \end{array} \right\} \tag{11}$$

where $cr^d(i)$ represents the critical child of node i with power degree d and $pa(v)$ represents the parent node of v. The assignment excludes ascendant nodes to prevent the occurrence of cycle in the network. The only exception is that we assign the parent of $cr^{d_{new}}(i)$ to i no matter whether $cr^{d_{new}}(i)$ belongs to *ascendant*(i) or not. If $cr^{d_{new}}(i) \in$ *ascendant*(i), the node k with $pa(k) = cr^{d_{new}}(i)$ is reassigned a parent by means of the minimal incremental power. This special case can also be shown in Fig. 5. In Fig. 5(b), we firstly reassign the parent of $cr^1(B)$ (i.e. node C) by removing link e_{sc}, e_{cb} and adding link e_{bc}. Then in Fig. 5(c), we reassign the parent of the node k with $pa(k) = cr^1(B) = C$ (i.e. reassign the parent of node B to node S).

Fig. 5 << Insert Fig. 5 about here >>

When the power degree of a node is decreased, those children nodes which cannot be reached any more will be disconnected. Each of these nodes is also reconnected by the node which requires the minimal incremental power. Examples of increasing and decreasing the power degree by one are shown in Fig. 6, which is based on Fig. 5(a). In Fig. 6(b), node D is the node (except S, the original parent of C) which needs the minimal incremental power to be the parent of C. In other words, if the power degree of a given node i is decreased from d_{old} to d_{new}, nodes in the set

$$\{cr^d(i) \mid d_{new} < d \le d_{old}\} \tag{12}$$

are re-assigned parents.

Fig. 6 << Insert Fig. 6 about here >>

3.5. Intensified r-shrink

We take r-shrink [21] as the local search algorithm in our PSO. It is applied in a Variable Neighborhood Descent (VND) [36] framework. The variable r in r-shrink governs the level of power shrinkage of a node. The variable r is the number of farthest children that will be removed from the children set of the node to attempt to reduce the transmission power level and power consumption. The r-shrink within VND process is given in Table 1. Here we do an improvement to the r-shrink based on some observations. Take Fig. 7 as an example. If node G is disconnected and is to be re-connected in r-shrink, we can re-connect it by node A or D without any additional incremental power. In the original r-shrink, there was no discussion on the choice of the parent node. Remind that we cannot assign a descendant node of a disconnected node to be the parent node of the disconnected one (since there will be a cycle). If we assign A as the parent node of G (Fig. 7(a)), G can serve as the parent node for five nodes in the shaded region in the upcoming r-shrink actions. However, if we assign D as the parent node of G (Fig. 7(b)), G can serve as the parent node for only two nodes. This observation leads to the idea: if we set the node with lower depth as the parent, we can generate more alternative solutions in the future r-shrink actions. Thus, when a node can be implicitly transmitted by more than one node without additional cost, it is transmitted by the node with the lowest depth in the broadcast tree. We call this improved version the *intensified r-shrink*.

Table 1 << Insert Table 1 about here >>

Fig. 7 << Insert Fig. 7 about here >>

3.6. The proposed PSO-based algorithm

A population of μ particles $x_1, x_2, \ldots, x_\mu$.

A set of μ velocity vectors $v_1, v_2, \ldots, v_\mu$.

The particle and neighborhood best-known solutions $x_1^{pb}, x_2^{pb}, \ldots, x_\mu^{pb}$ and $x_1^{nb}, x_2^{nb}, \ldots, x_\mu^{nb}$.

The algorithm works as follows:

Step 1 Initialization

1.1 For $i = 1\ldots\mu$, randomly build a broadcast tree for x_i, and then apply the intensified r-shrink within the VND for x_i.

1.2 For $i = 1\ldots\mu$, $x_i^{pb} = x_i$.

1.3 For $i = 1\ldots\mu$, choose the best of personal best solution x_j^{pb} of K neighbors j as x_i^{nb}.

1.4 For $i = 1\ldots\mu$, assign random values between $[-v_{max}, v_{max}]$ for all values in v_i.

Step 2 Landing

2.1 Repeat **2.1.1** to **2.1.3** for $i = 1\ldots\mu$.

2.1.1 Calculate the new power degree by (8).

2.1.2 Adjust the broadcast tree according to the new power degrees.

2.2 For $i = 1\ldots\mu$, update neighbors according to the distance between x_i and x_j^{pb} $\forall j \neq i$.

2.3 Update x_i^{nb} by choosing the best solution from K neighbors.

Step 3 Acceleration, Craziness, and Local Search

3.1 Acceleration: For $i = 1\ldots\mu$, calculate velocity v_i for the next iteration by (9). Then, confine velocity values by

$$v_i = \max(-v_{max}, \min(v_{max}, v_i)) \tag{13}$$

3.2 Craziness: For $i = 1\ldots\mu$, assign random values in $[-v_{max}, v_{max}]$ for a random element in v_i with probability p_c.

3.3 Local Search: For $i = 1\ldots\mu$, apply the intensified r-shrink within the VND procedure for x_i.

Step 4 Update

4.1 For $i = 1\ldots\mu$, Compute fitness $f(x_i)$ by (3), and update $x_i^{pb} = x_i$ if $f(x_i) < f(x_i^{pb})$.

4.2 For $i = 1\ldots\mu$, if $\exists\, j \neq i,\ x_i^{pb} = x_j^{pb}$, randomly initialize x_j and update $x_j^{pb} = x_j$.

4.3 If the stopping criterion is met, then terminate the search. Otherwise, return to **Step 2**.

In step 4.2, we replace the particle j by a randomly generated particle if we find that x_j^{pb} is identical to x_i^{pb} in order to increase the population diversity. In this condition, the personal-best solution of this newly introduced random particle is set to itself ($x_j^{pb} = j$).

4. Extension to the dynamic MEB problem

WANETs must be scalable and have the ability to sustain in a harsh environment. Robustness and fault tolerance are essential characteristics, and it is important to address the issue of node failure. Nonetheless, there was a noticeable absence of research dealing with node destabilization in the MEB problem. In order to help fill this gap, this section looks into a solution for the MEB problem in the dynamic environment.

4.1. Repairing scheme

In unstable and rapidly changing WANETs, it may not be possible to update the network topology to the optimal solution for large-scale instances. In addition to the computation time of the algorithm, the communication overhead of topology changes must be taken into account, too. To make the solution scalable and feasible, it is probably better to re-connect the disconnected nodes while keeping the existing links.

Here, we present a repairing scheme for the dynamic MEB problem. Two scenarios are considered, node removal and node insertion. The former deletes some nodes in the network, simulating the circumstances of node failure; the latter adds some new nodes into the network, simulating the scenarios of discovery of new nodes or transforming states of some nodes from inactive to active. Fig. 8(a) shows the broadcast tree before node failure. In Fig. 8(b), node A is detected failed, and then node C and the subtree which is rooted at node B are disconnected. Let V_c denote the set of nodes in the broadcast tree and V_u denote the set of root nodes of disconnected components. In the example, node B and node C form the set V_u. An example of the network after repairing is shown in Fig. 8(c). Note that the descendant nodes of B are neither in V_c nor V_u, and they will join V_c after node B is connected. Our task is to find a constructive heuristic to re-connect the disconnected parts effectively and efficiently.

Fig. 8 << Insert Fig. 8 about here >>

4.2. Conditional Incremental Power (CIP)

Here, we present a simple heuristic called Conditional Incremental Power (CIP). The detailed algorithm is given in Table 2. Different from the existing heuristics, CIP takes into account not only the nodes in V_c but also the nodes in V_u when re-connecting the nodes. It firstly checks the minimum incremental power (τ_i) required for each uncovered node i to be covered by nodes in $V \backslash V_c$ except itself. After that, it calculates the minimum incremental power (τ_i') required for node i to be covered by nodes in V_c. When choosing the new-joining node from the candidate links, CIP gives the links with $\tau_i < \tau_i'$ lower priorities. If there is more than one node that has the same priority, then CIP connects the node with the minimum incremental power.

Table 2 << Insert Table 2 about here >>

The concept of CIP is quite simple. In existing simple heuristics, an uncovered node is usually chosen as new-joining node if it is close to V_c. Nonetheless, it doesn't consider the situation that there might be another path which can produce lower incremental power, but the path is just not available because it has to go through some nodes which are not in V_c yet. Especially in the case of the network which runs into node failure, there are a lot of nodes in the disconnected components. It is very possible that the nodes in the disconnected components are offering better choices to connect the uncovered nodes. Consequently, CIP prefers not to choose the nodes which have τ_i lower than τ_i', because the paths which produce lower incremental power may appear later by joining other disconnected components.

Fig. 9 illustrates the integrated approach of PSO and CIP to solve the dynamic MEB problem. Given a network, we first generate the broadcast tree by PSO-based optimization. When there is some change in the network, we can use the CIP to rebuild the broadcast tree efficiently and properly. As the network changes more and more, using only the CIP may not be able to maintain very high solution quality. At this time, we can run the PSO optimization method again (in the background) to improve the solution.

Fig. 9 << Insert Fig. 9 about here >>

5. Experimental results

5.1. Benchmark instances

The experiments are conducted using instances with different network sizes. The benchmark instances considered in the experiments are from DAG [37] and ALBCOM [29]. In these instances, nodes are randomly scattered in a 1000×1000 square. The numbers of nodes are 20, 50, and 100, and there are 30 instances for each network size. The path loss exponent α is 2, and the power threshold ξ is normalized to 1.

5.2. Benchmark algorithms

We compared our proposed algorithm with several state-of-the-art approaches, including ELS (Evolutionary Local Search) [27], HGA (Hybrid Genetic Algorithm) [28], and ACO (Ant Colony Optimization) [29]. The compared approaches are innovative and well known at solving the static MEB problem, and they show the most effective performance in terms of solution quality. We also took the well-known heuristic BIP [14] as a referenced algorithm. All of the compared algorithms are listed in Table 3. Most of these algorithms combined the local search procedure into their search schemes. Therefore, local search would play an important role in solving the MEB problem. In the original work of r-shrink [21], the authors only took 1-shrink as an example, and the detailed implementation when the r value is greater than one is ambiguous. Hence, in most compared algorithms, they implemented their own r-shrink procedures. The parameter settings of all benchmark algorithms are identical to those in the original papers and are listed in Table 4. ELS and HGA have four parameters, and ACO has ten parameters.

Table 3 << Insert Table 3 about here >>

Table 4 << Insert Table 4 about here >>

5.3. Parameter setting

Our proposed PSO algorithm contains six parameters, and they were fine-tuned by hand and based on suggestions from literatures [38, 39], where the authors recommended a proper ranges of inertia weight and weights for accelerations toward *pbest* and *nbest*. Complete parameter settings and simulation setups are shown in Table 5. The algorithms were coded in C++ programming language and run on a personal computer with Intel® 3.2 GHz CPU and 2GB RAM. We followed the stopping criterion in [29] and set the CPU time limit accordingly, which is given in Table 6. Each algorithm was applied to solve each instance for 30 times.

Table 5 << Insert Table 5 about here >>

Table 6 << Insert Table 6 about here >>

5.4. Performance evaluation in the static MEB problem

5.4.1. 20-node instances

The proposed PSO-based approach can solve small-scale instances in a very short time. For comparative performance in different scales, the experimental results of 20-node instances are shown in Table 7. Optimal solutions are found for all instances in all runs. The average computation time is 0.05 second.

Table 7 << Insert Table 7 about here >>

5.4.2. 50-node instances

The detailed experimental results for 50-node instances are shown in Table 8. There are 30 instances, and the values of the optimal solutions are obtained from [27]. For each algorithm, we present the average deviation percentage from the optimal solutions and the times of finding optimal solutions among 30 runs. Note that the average computation time refers to the average time required to find the optimal solutions, and the trials which fail to find the optimum are not included.

Table 8 << Insert Table 8 about here >>

Among the four algorithms, ELS performs the worst in terms of both solution quality and computation time. HGA performs slightly better than ELS, but there is still a gap from our proposed PSO-based approach. HGA spends 13.71 seconds on average to find the optimal solutions, but PSO only needs 1.45 seconds. The computation time is only for reference since the computing platform in the experiments in [28] was different. They used Intel® 3.2 GHz CPU and 512 MB RAM. A potential weakness of HGA is the permutation encoding, which can only represent the solutions before local search. The broadcast tree after local search cannot be encoded back to a permutation; therefore, only Baldwinian type [40] of local search is applicable.

5.4.3. 100-node instances

The previous experiment shows that only ACO matches our proposed PSO in solving 50-node instances. ACO is an approach imitating the food foraging behavior of ants. It puts pheromone on the links between nodes. The intensity of pheromone was updated according to the quality of solutions comprising these links, and the intensity affects the selection of links to form the broadcast network. In addition, ACO used *r*-shrink as the local search procedure. We compared our proposed algorithm with ACO in solving 100-node instances, and the experimental results are given in Table 9.

Table 9 << Insert Table 9 about here >>

Because it takes tremendous time for obtaining the optimal solutions by the linear programming solver, only the best-known solutions are presented in Table 9. The experimental results show that the proposed PSO outperforms ACO in terms of average solution quality, but it also takes more computation time. The power consumption of solutions found by ACO exceeds that of the best-known solutions 0.21% on average, and our PSO exceeds only by 0.07%. The average computation time of ACO and PSO is 28.7 and 40.41 seconds, respectively. In terms of the average solution quality, PSO is better than ACO for 16 instances and equivalent for 6 instances. The independent *t*-test was conducted to examine the differences. The proposed PSO is statistically better than ACO for 12 instances and statistically worse for only 2 instances. The two algorithms have statistically equal results in the remaining 16 instances.

5.4.4. Intensified *r*-shrink

In this subsection we analyze the effectiveness of the intensified *r*-shrink. Table 10 presents some results for the comparison of *r*-shrink and intensified *r*-shrink. We found that intensified *r*-shrink improves the BIP solution slightly better than *r*-shrink does (11.49% vs. 11.85% in terms of average excess percentage). (BIP + intensified *r*-shrink) generates better solutions for half problem instances and equal solutions for the other half. Although the improvement of power consumption is not large, the search process can benefit a lot from the intensified *r*-shrink. In the search process, the intensified *r*-shrink is applied to every particle in the population. Therefore, the accumulated improvement will be substantial. The average excess over the best known solutions is 0.42% and 0.07% for (PSO + *r*-shrink) and (PSO + intensified *r*-shrink), respectively. (PSO + intensified *r*-shrink) outperforms the other for 27 out of 30 100-node instances.

Table 10 << Insert Table 10 about here >>

The broadcast trees built by (BIP + *r*-shrink) and our proposed (PSO + intensified *r*-shrink) for instance p.50.02 and p.100.07 are shown in Fig. 10 and Fig. 11. Both of them applied *r*-shrink with VND procedure. The specified source node is denoted as the gray node. Fig. 11 shows that these two algorithms result in very different structures for instance p100.07. BIP tends to transmit packets by using multiple short intermediate transmissions, but our proposed PSO can find more inexpensive way, which forwards packets with few long-distance transmissions covering a lot of nodes. Although the proposed PSO can fully utilize the Wireless Multicast Advantage (WMA) to minimize the total energy consumption, one potential problem is the imbalance between the energy consumption of the nodes. In Fig. 11, node 38 consumes much more energy than other nodes do. In practical conditions, the node may not have such amount of energy. Besides, this may cause this node to run out of energy quickly. A possible solution is to add the constraint of the maximum energy consumption for each node in the problem formulation. We leave testing of the proposed PSO in the extended problem as our future work.

Fig. 10 << Insert Fig. 10 about here >>

Fig. 11 << Insert Fig. 11 about here >>

5.5. Performance evaluation for the dynamic MEB problem

5.5.1. Benchmark instances and algorithms

We created the problem instances for the dynamic MEB problem based on the instances in DAG [37] and ALBCOM [29]. The experiments can be categorized into two parts: *node removal* and *node insertion*. In the former scenario, we sequentially removed the nodes in the network in the order of node ID respectively. For example, if five nodes in an n-node network are going to be deleted, the nodes set {0, 1, 2, 3, 4}, {5, 6, 7, 8, 9}, …, and $\{n-5, n-4, n-3, n-2, n-1\}$ were deleted independently in each problem instance. We will pass over the source node and delete the next node. As for the node insertion scenario, we create the instances by adding randomly deployed nodes. We removed/inserted 10%, 20%, and 50% of nodes in the problem instances with 50 and 100 nodes. We compared the performance of CIP with five simple heuristics: ABC [17], BIP [14], GPBE [15], MST [14], and SPT [14]. We will show the experimental results in terms of average deviation percentage.

5.5.2. Node removal

The experimental results of node removal are shown in Fig. 12. The denotation $n-a$ stands for the n-node instances with a

nodes removed from the network. SPT obtained the worst result, which has average deviation percentage about 22.3%, and we exclude the results of SPT for succinctness in Fig. 12. CIP returns better results than all benchmark algorithms. Although the deviation percentage of CIP rises when the number of deleted nodes increases, the gap between CIP and other algorithms also increases. It implies that CIP saves much more energy than other heuristics when the network is more unstable. All heuristics can repair the network topology in very short time, and the computation time is always less than 0.01 second for all tested instances. Thus, we omit the computation time of all experiments in this and the following subsections.

Fig. 12 << Insert Fig. 12 about here >>

5.5.3. Node insertion

The experimental results of node insertion are shown in Fig. 13. The denotation $n+a$ stands for n-node instances with a nodes inserted into the network. In order to differentiate the gaps between algorithms, the vertical axis is presented by logarithmic scale.

Fig. 13 << Insert Fig. 13 about here >>

As shown in Fig. 13, GPBE produces the best results in some cases when the ratio of inserted nodes is small. However, as the number of inserted nodes increases, the performance of GPBE is getting worse. Algorithms ABC, BIP, GPBE, and the proposed CIP have average power consumption higher than the best solution by 0.06%, 0.04%, 0.07%, and 0.03%, respectively. The deviation percentage of MST is up to 15%, and that of SPT is up to 115%. CIP is again the best performer.

5.5.4. 100-node instances

To further study the effectiveness of the proposed CIP algorithm, we also compared the performance of six algorithms in constructing the complete broadcast trees for 100-node instances [29]. The detail results are given in Table 11.

Table 11 << Insert Table 11 about here >>

Our proposed CIP returns the best result, which has the deviation percentage from the best solution by about 1%. ABC, BIP, GPBE, MST, and SPT have the deviation percentage from the best solution by about 5%, 4%, 8%, 11%, and 37% respectively. CIP also obtains the best results in 13 out of 30 instances. This evidence shows that CIP is not only capable of repairing the disconnect components, but it is also a fine heuristic for constructing broadcast trees.

6. Conclusion

The overall power consumption of WANETs can be prominently improved through the development of a better routing strategy. In this paper, we proposed an algorithm based on PSO for solving the MEB problem in WANETs. We transformed broadcast trees to encoded sequences in search space by the proposed power degree encoding. An intensified *r*-shrink procedure was proposed based on the analysis of node depth to serve as a local search procedure in the PSO algorithm. We also developed a simple and effective heuristic, CIP, for the dynamic scenarios. Experimental results show that our proposed PSO algorithm outperforms three algorithms for 60 instances with 50 nodes and 100 nodes in the static environments. The CIP also outperforms five algorithms for instances with various network sizes.

There is a continuing need for adequate modifications toward the practical applications of WANET. As the experimental results indicated, the next step is to take into account the constraint of the maximum energy consumption of a single node. In addition to taking this factor as a constraint, we can consider objectives such as minimizing the maximum energy consumption or minimizing the imbalance between energy consumption of nodes. The existence of more than one objective will turn our goal to seek for the Pareto optimal solutions. On the application side, this makes the problem more realistic; on the algorithm side, this also offers an opportunity of designing and testing multiobjective PSO [41][42]. Several other factors including omni-directional antennas, multicast scenarios, heterogeneous space, and the power needed to process and reroute information in intermediary nodes should also be added into the problem. The proposed algorithm is to be tested in these extended problems and is to be enhanced for better performance.

Acknowledgment

The authors are very grateful to the editor and anonymous reviewers for their valuable and instructive comments.

References

[1] X.M. Hu, J. Zhang, Y. Yu, H.S.H. Chung, Y.L. Li, Y.H. Shi, X.N. Luo, Hybrid genetic algorithm using a forward encoding scheme for lifetime maximization of wireless sensor networks, IEEE Transactions on Evolutionary Computation 14 (2010) 766-781.

[2] C.-C. Lai, C.-K. Ting, R.-S. Ko, An effective genetic algorithm to improve wireless sensor network lifetime for large-scale surveillance applications, in: Proceedings of IEEE Congress on Evolutionary Computation, 2007, pp. 3531-3538.

[3] M. Younis, K. Akkaya, Strategies and techniques for node placement in wireless sensor networks: A survey, Ad Hoc Networks 6 (2008) 621-655.

[4] J.-A. Jiang, T.-S. Lin, C.-L. Chuang, C.-P. Chen, C.-H. Sun, J.-Y. Juang, J.-C. Lin, W.-W. Liang, A QoS-guaranteed coverage precedence routing algorithm for wireless sensor networks, Sensors 11 (2011) 3418-3438.

[5] S. Guo, O.W.W. Yang, Energy-aware multicasting in wireless ad hoc networks: A survey and discussion, Computer Communications 30 (2007) 2129-2148.

[6] C. Miller, C. Poellabauer, A decentralized approach to minimum-energy broadcasting in static ad hoc networks, in: P. Ruiz, J. Garcia-Luna-Aceves (Eds.) Ad-Hoc, Mobile and Wireless Networks, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2009, pp. 298-311.

[7] M. Čagalj, J.-P. Hubaux, C. Enz, Minimum-energy broadcast in all-wireless networks: NP-completeness and distribution issues, in: Proceedings of the 8th Annual International Conference on Mobile Computing and Networking, 2002, pp. 172-182.

[8] W. Liang, Constructing minimum-energy broadcast trees in wireless ad hoc networks, in: Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing, Lausanne, Switzerland, 2002, pp. 112-122.

[9] P.-C. Hsiao, T.-C. Chiang, L.-C. Fu, Particle Swarm Optimization for the Minimum Energy Broadcast Problem in Wireless Ad-Hoc Networks, in: Proceedings of IEEE Congress on Evolutionary Computation, Brisbane, Austrailia, 2012, pp. 3114-3121.

[10] A. Konstantinidis, K. Yang, H.-H. Chen, Q. Zhang, Energy-aware topology control for wireless sensor networks using memetic algorithms, Computer Communications 30 (2007) 2753-2764.

[11] J. Bauer, D. Haugland, D. Yuan, A fast local search method for minimum energy broadcast in wireless ad hoc networks, Operations Research Letters 37 (2009) 75-79.

[12] I. Kang, R. Poovendran, Iterated local optimization for minimum energy broadcast, in: Proceedings of International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, 2005, pp. 332-341.

[13] M. Cardei, J. Wu, Energy-efficient coverage problems in wireless ad-hoc sensor networks, Computer Communications 29 (2006) 413-420.

[14] J.E. Wieselthier, G.D. Nguyen, A. Ephremides, On the construction of energy-efficient broadcast and multicast trees in wireless networks, in: Proceedings of IEEE International Conference on Computer Communications, 2000, pp. 585-594.

[15] I. Kang, R. Poovendran, A novel power-efficient broadcast routing algorithm exploiting broadcast efficiency, in: Proceedings of IEEE Vehicular Technology Conference, 2003, pp. 2926-2930.

[16] P.J. Wan, G. Călinescu, X.Y. Li, O. Frieder, Minimum-energy broadcasting in static ad hoc wireless networks, Wireless Networks 8 (2002) 607-617.

[17] R. Klasing, A. Navarra, A. Papadopoulos, S. Pérennes, Adaptive Broadcast Consumption (ABC), a new heuristic and new bounds for the minimum energy broadcast routing problem, in: N. Mitrou, K. Kontovasilis, G. Rouskas, I. Iliadis, L. Merakos (Eds.) Networking 2004, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2004, pp. 866-877.

[18] M.X. Cheng, J. Sun, M. Min, D.-Z. Du, Energy-efficient broadcast and multicast routing in ad hoc wireless networks, in: Proceedings of IEEE International Performance, Computing, and Communications Conference, 2003, pp. 87-94.

[19] M.X. Cheng, J. Sun, M. Min, Y. Li, W. Wu, Energy-efficient broadcast and multicast routing in multihop ad hoc wireless networks, Wireless Communications and Mobile Computing 6 (2006) 213-223.

[20] L. Lin, W. Qiwu, A label based greedy algorithm for minimum energy consumption multicast routing in ad hoc networks, in: Proceedings of IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, 2011, pp. 113-117.

[21] A.K. Das, R.J. Marks, M. El-Sharkawi, P. Arabshahi, A. Gray, r-shrink: a heuristic for improving minimum power broadcast trees in wireless networks, in: Proceedings of IEEE Global Telecommunications Conference, 2003, pp. 523-527.

[22] S. Guo, O. Yang, A dynamic multicast tree reconstruction algorithm for minimum-energy multicasting in wireless ad hoc networks, in: Proceedings of IEEE International Conference on Performance, Computing, and Communications, 2004, pp. 637-642.

[23] I. Kang, R. Poovendran, Broadcast with heterogeneous node capability, in: Proceedings of IEEE Global Telecommunications Conference, 2004, pp. 4114-4119.

[24] A. Konstantinidis, K. Yang, Q. Zhang, D. Zeinalipour-Yazti, A multi-objective evolutionary algorithm for the deployment and power assignment problem in wireless sensor networks, Computer Networks 54 (2010) 960-976.

[25] G. Molina, E. Alba, Location discovery in Wireless Sensor Networks using metaheuristics, Applied Soft Computing 11 (2011) 1223-1240.

[26] J.-A. Jiang, C.-P. Chen, C.-L. Chuang, T.-S. Lin, C.-L. Tseng, E.-C. Yang, Y.-C. Wang, CoCMA: Energy-efficient coverage control in cluster-based wireless sensor networks using a memetic algorithm, Sensors 9 (2009) 4918-4940.

[27] S. Wolf, P. Merz, Evolutionary local search for the minimum energy broadcast problem, in: J. van Hemert, C. Cotta (Eds.) Proceedings of the 8th European Conference on Evolutionary Computation in Combinatorial Optimisation, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2008, pp. 61-72.

[28] A. Singh, W.N. Bhukya, A hybrid genetic algorithm for the minimum energy broadcast problem in wireless ad hoc networks, Applied Soft Computing 11 (2011) 667-674.

[29] H. Hernández, C. Blum, Ant colony optimization for multicasting in static wireless ad-hoc networks, Swarm Intelligence 3 (2009) 125-148.

[30] H. Hernández, C. Blum, Minimum energy broadcasting in wireless sensor networks: An ant colony optimization approach for a realistic antenna model, Applied Soft Computing 11 (2011) 5684-5694.

[31] H. Hernández, C. Blum, G. Francès, Ant colony optimization for energy-efficient broadcasting in ad-hoc networks, in: M. Dorigo, M. Birattari, C. Blum, M. Clerc, T. Stützle, A. Winfield (Eds.) Proceedings of International Conference on Ant Colony Optmization and Swarm Intelligence, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2008, pp. 25-36.

[32] H. Hernández, C. Blum, Distributed ant colony optimization for minimum energy broadcasting in sensor networks with realistic antennas, Computers & Mathematics with Applications (2012).

[33] A.K. Das, R.J. Marks, M. El-Sharkawi, P. Arabshahi, A. Gray, The minimum power broadcast problem in wireless networks: an ant colony system approach, in: Proceedings of IEEE CAS Workshop on Wireless Communications and Networking, Pasadena, CA, 2002.

[34] W. Xiang, W. Xinheng, L. Rui, Solving minimum power broadcast problem in wireless ad-hoc networks using genetic algorithm, in: Proceedings of Communication Networks and Services Research Conference, 2008, pp. 203-207.

[35] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Networks, 1995, pp. 1942-1948.

[36] P. Hansen, N. Mladenović Variable neighborhood search, Handbook of Metaheuristics (2003) 145-184.

[37] S. Al-Shihabi, P. Merz, S. Wolf, Nested partitioning for the minimum energy broadcast problem, in: V. Maniezzo, R. Battiti, J.-P. Watson (Eds.) Proceedings of Learning and Intelligent Optimization, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2008, pp. 1-11.

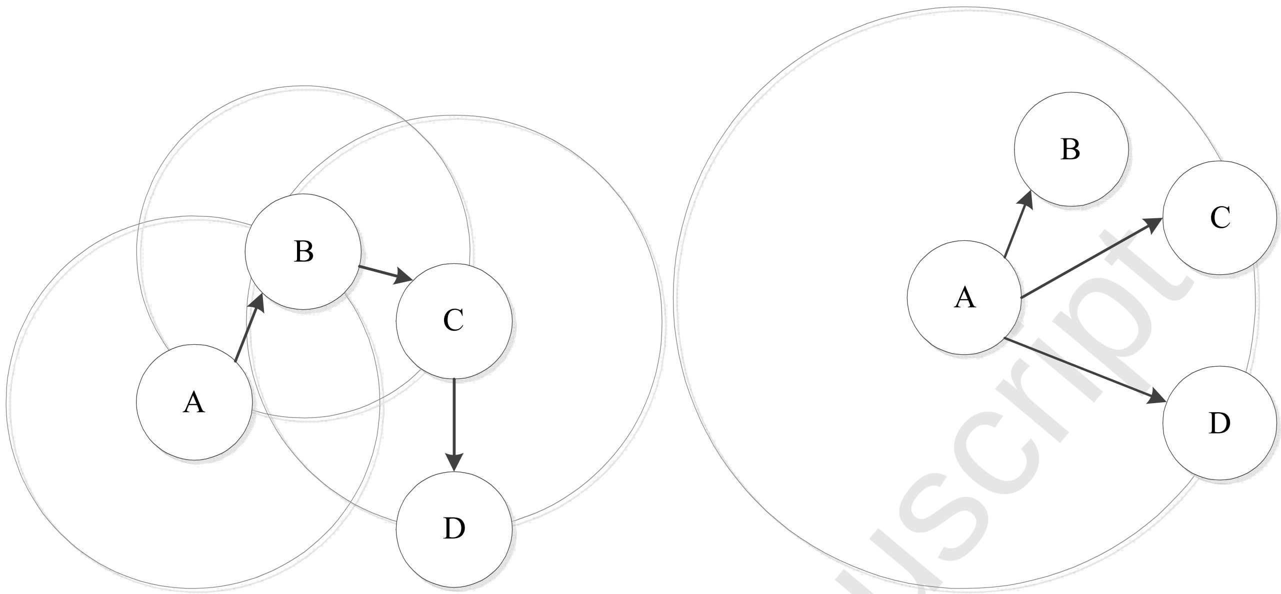
[38] Y. Shi, R. Eberhart, Parameter selection in particle swarm optimization, in: V. Porto, N. Saravanan, D. Waagen, A. Eiben (Eds.) Evolutionary Programming VII, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 1998, pp. 591-600.

[39] M. Clerc, J. Kennedy, The particle swarm - explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6 (2002) 58-73.

[40] F. Gruau, D. Whitley, Adding learning to the cellular development of neural networks: Evolution and the Baldwin effect, Evolutionary Computation 1 (1993) 213-233.

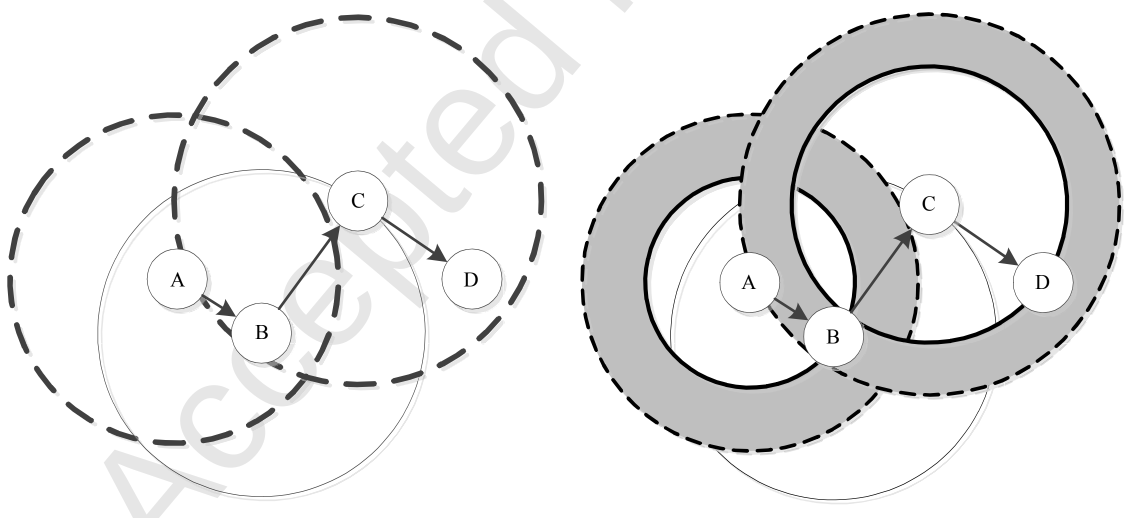
[41] C. A. C. Coello, G. T. Pulido, M. S. Lechuga, Handling multiple objectives with particle swarm optimization, IEEE Transactions on Evolutionary Computation 8 (2004) 256-279.

[42] M. Reyers-Sierra, C. A. C. Coello, Multi-objective particle swarm optimization: a survey of the state-of-the-art, International Journal of Computational Intelligence Research, 2 (2006) 287 - 308.



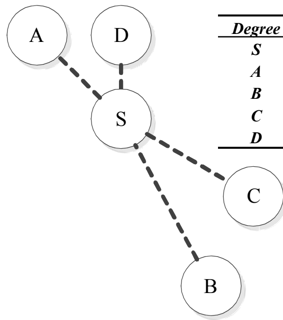
(a) Transmission by multiple hops. (b) Transmission by a single long hop.

Fig. 14 The characteristic of WMA. All nodes within the transmission range can receive the packets.



(a) Broadcast tree decoded by permutation sequence (b) Broadcast tree after reduction procedure.

Fig. 15 Transmission power level reduction for power level encoding. The gray area is the wasted power before reduction.



Degree	*0*	*1*	*2*	*3*	*4*
S	S	D	A	C	B
A	A	D	C	B	
B	B	C	D	A	
C	C	B	D	A	
D	D	A	C	B	

Fig. 16 The power degree table: in each row starting by a certain node i, the nodes are sorted in non-decreasing order of distance from i. For node S, for example, the sequence of nodes from the nearest to the farthest is $\{D, A, C, B\}$. If S transmits to D, its power degree is 1.

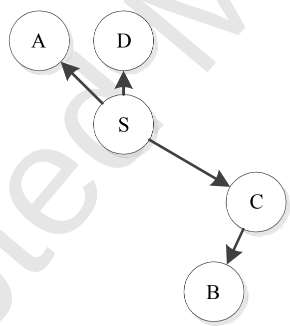
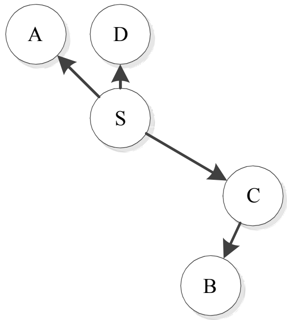
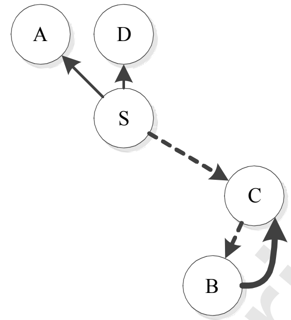


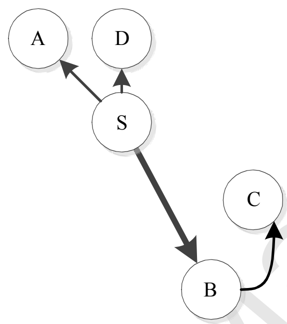
Fig. 17 The power degree encoding for a specific broadcast tree.
In this case, the corresponding chromosome is {3, 0, 0, 1, 0}.



(a) The original broadcast tree.

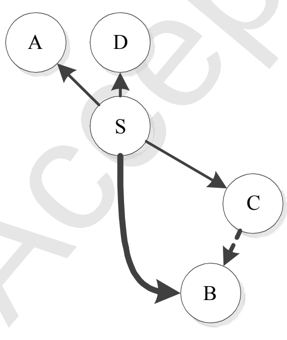


(b) Reassign the parent of *C* (add e_{BC}).

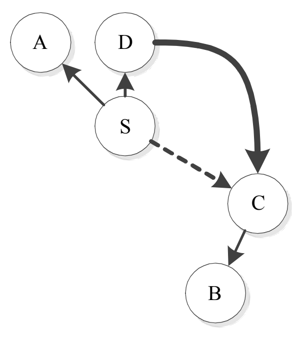


(c) Reconnect *B* with the node (*S*) which brings the minimum incremental power.

Fig. 18 The special case of increasing power degree. It is assumed that the power degree of *B* in (a) increases by one.

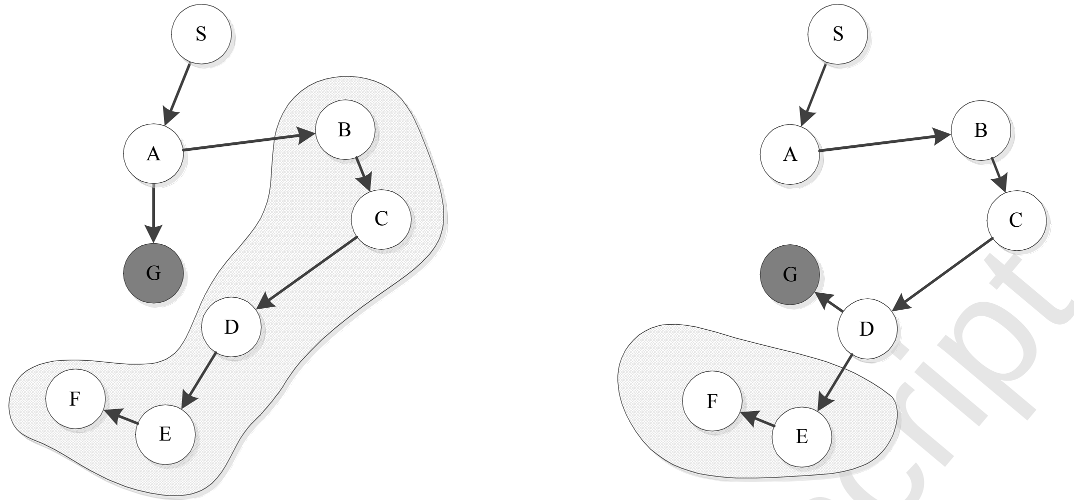


(a) Increase power degree.



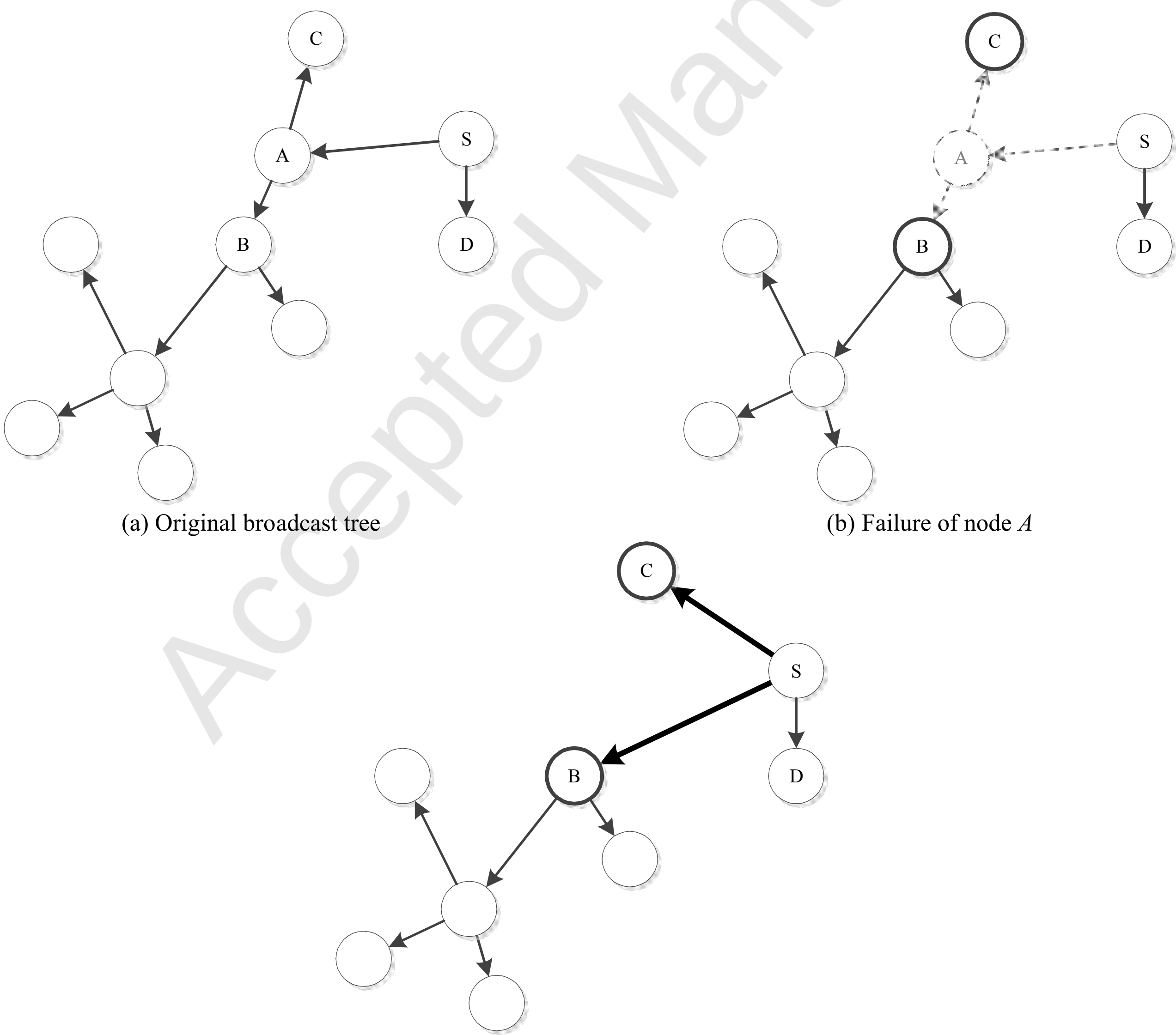
(b) Decrease power degree.

Fig. 19 Illustration for increasing and decreasing power degree based on Fig. 5(a).
(a) Increase the power degree of *S* by one. (b) Decrease the power degree of *S* by one.



(a) A is the parent of G. (b) D is the parent of G.

Fig. 20 Illustration of deciding the parent of G, which can be implicitly covered by A and D.



(a) Original broadcast tree (b) Failure of node A

(c) Reconnect the disconnected nodes B and C.

Fig. 21 The scenario of node failure. If A fails, the children of A (node B and C) are in the set V_u.

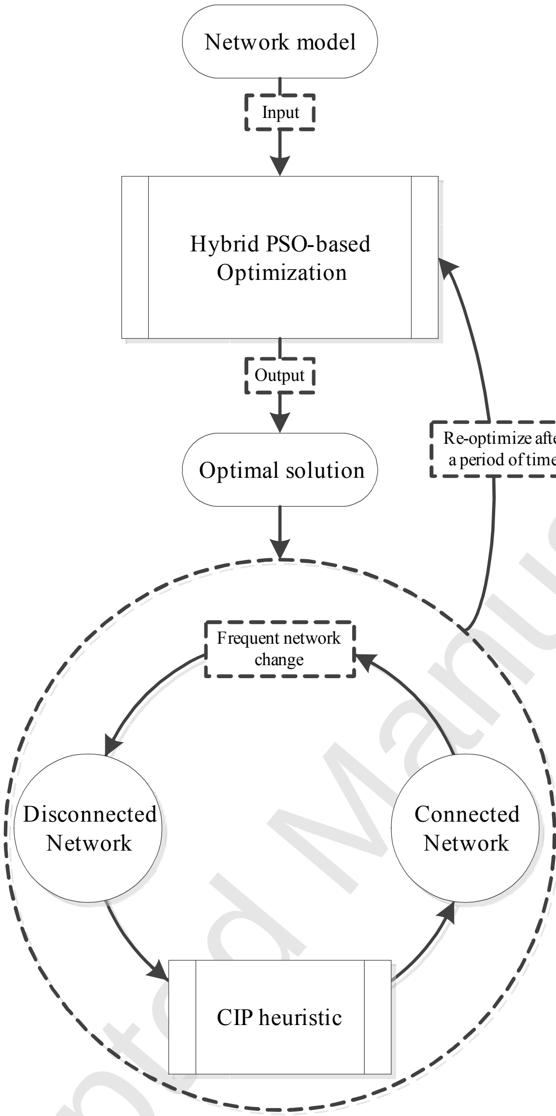
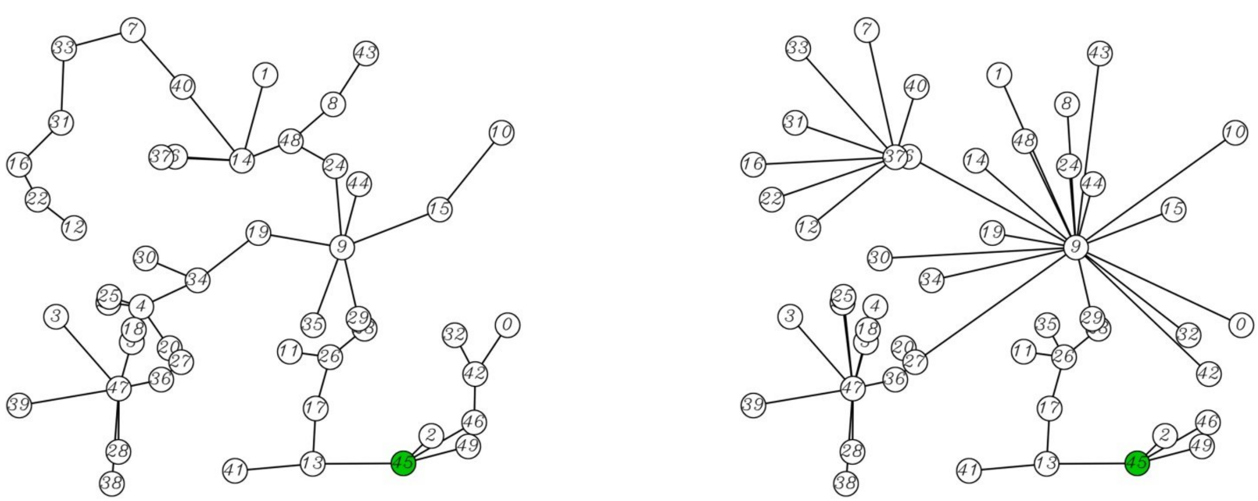


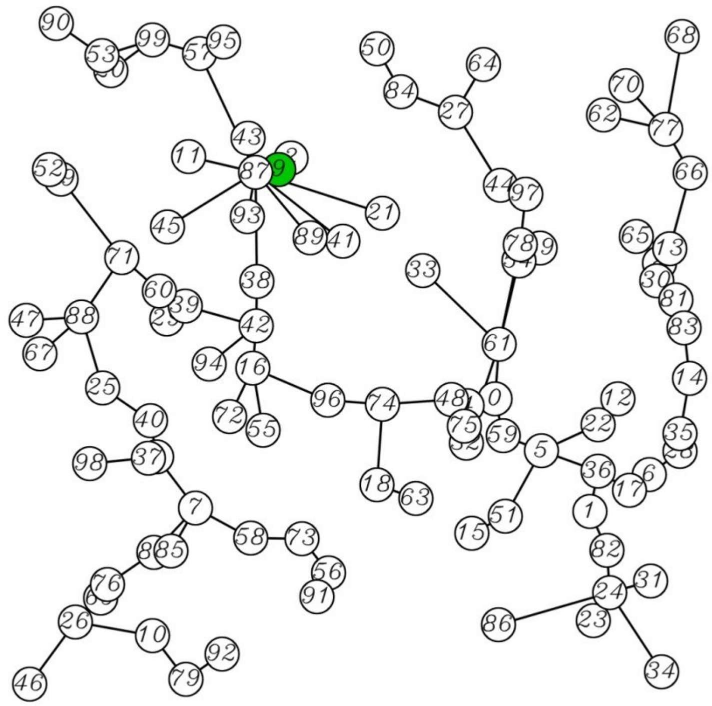
Fig. 22 Framework of the integrated approach to the dynamic MEB problem.



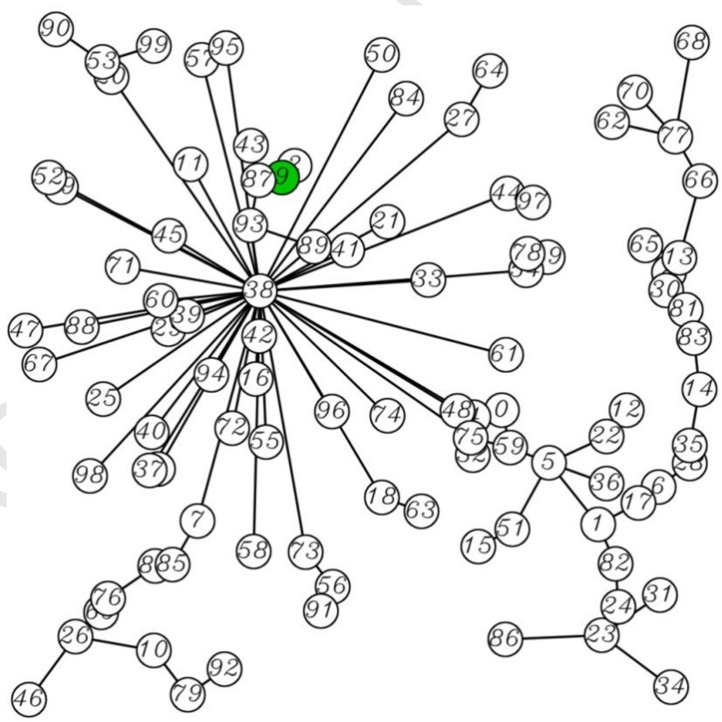
(a) BIP + *r*-shrink (VND)

(b) Proposed PSO + intensified *r*-shrink (VND)

Fig. 23 The broadcast tree built by (a) BIP and (b) proposed PSO-based approach in instance p.50.02.



(a) BIP + *r*-shrink (VND)



(b) Proposed PSO + intensified *r*-shrink (VND)

Fig. 24 The broadcast tree built by (a) BIP and (b) proposed PSO-based approach in instance p.100.07.

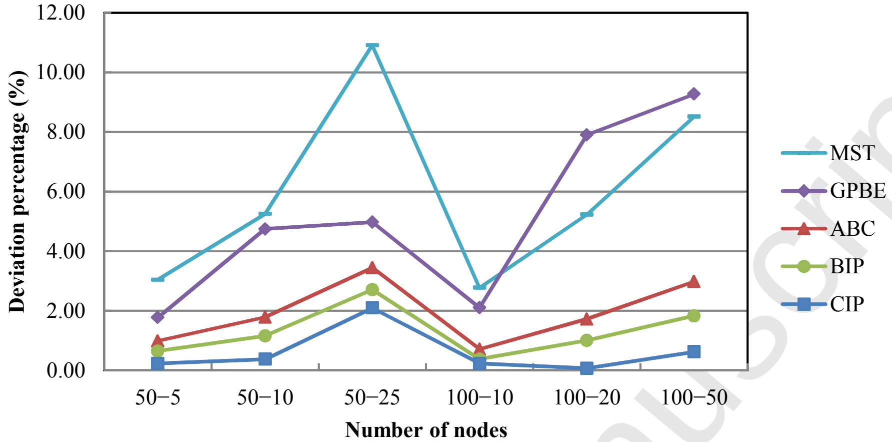


Fig. 25 Experimental results of node removal.

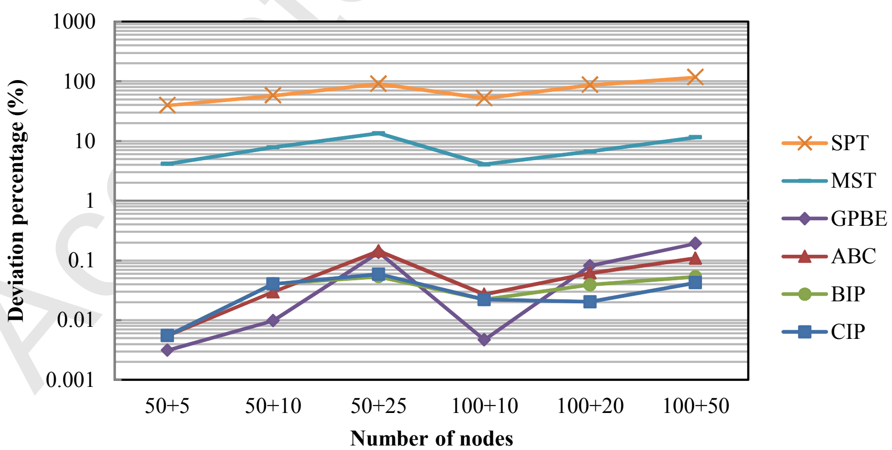


Fig. 26 Experimental results of node insertion.

Table 12 r-shrink within the VND procedure.

Set $r \leftarrow 1$
repeat
  $x' \leftarrow r$-shrink(x)
  if $f(x') < f(x)$ **then**
    $x \leftarrow x'$
    $r \leftarrow 1$
  else
    $r \leftarrow r + 1$
  end if
until $r > r_{max}$

Table 13 The algorithm of Conditional Incremental Power (CIP).

Notation	Meaning
V	the set of nodes in the network
V_c	the set of nodes in the broadcast tree
V_u	the set of root nodes of disconnected components
τ_i	the minimum incremental power when linking up node i from one in $V \setminus V_c$
τ'_i	the minimum incremental power when linking up node i from one in V_c
ns	the set of nodes which has greater τ_i than τ'_i
$pair\_join$	a pair of nodes which is the link that is going to be joined to the tree

Input: the node set V of the wireless network, a specified source node $s \in V$, and the current broadcast tree $T = (V_c, E_c)$
Output: T, the broadcast tree which includes all nodes in the network

while $V_c \neq V$ **do**
  for $i \in V_u$ **do**
    Calculate τ_i and τ'_i
  end for
  $ns \leftarrow \{i \mid i \in V_u \wedge \tau'_i \leq \tau_i\}$
  if $ns \neq \emptyset$ **then**

$b \leftarrow$ the node $\in ns$ and has minimum τ'_i

$a \leftarrow$ the node $\in V_c$ and produce τ'_b after linking up b

$pair\_join \leftarrow (a,b)$

else

$b \leftarrow$ the node $\in V_u$ and has minimum τ'_i

$a \leftarrow$ the node $\in V_c$ and produce τ'_b after linking up b

$pair\_join \leftarrow (a,b)$

end if

$V_c \cup \{pair\_join.second\}$

$E_c \cup \{pair\_join\}$

end while

return $T = (V_c, E_c)$

Table 14 Benchmark algorithms for the static MEB problem

Algorithm	Authors	Type	Local search
BIP [13]	Wieselthier *et al.*	Simple Heuristic	-
ELS [26]	Wolf and Merz	Evolutionary algorithm	Modified *r*-shrink
HGA [27]	Singh and Bhukya	Evolutionary algorithm	Modified *r*-shrink
ACO [28]	Hernández and Blum	Evolutionary algorithm	*r*-shrink
Proposed PSO	Hsiao *et al.*	Evolutionary algorithm	Intensified *r*-shrink

Table 15 Parameter settings for the benchmark algorithms

Algorithm	Notation	Value	Meaning
ELS [27]	λ	500	population size
	r	1	*r* value for *r*-shrink
	β_{start}	n	initial mutation rate
	$\beta_{decrease}$	10%	mutation reduction rate
HGA [28]	*population*	600	population size
	r_{max}	2	maximum *r* value for *r*-shrink
	p_{better}	0.8	rate of better fitness is selected
	p_m	0.75	mutation rate
ACO [29]	[illegible]$_{max}$	0.99	maximum pheromone value
	[illegible]$_{min}$	0.01	minimum pheromone value
	ρ	0.1	learning rate
	r_{max}	$n-2$	maximum *r* value for *r*-shrink

c	8	number of candidates
n_a	10	number of ants
□□$_{ib}$	2/3,1/3,0,0	weight of iteration-best
□□$_{rb}$	1/3,2/3,1,0	weight of restart-best
□□$_{bs}$	0,0,0,1	weight of best-so-far
cf	0.7,0.9	convergence factor interval

n: the number of nodes

Table 16 Parameter setting for the proposed approach and simulation setup.

Notation	**Value**	**Meaning**
μ	40	population size
w	0.8	inertia weight
v_{max}	$n/4$	maximum value of velocity
p_c	0.05	crazy rate
K	4	neighborhood size
r_{max}	$n-2$	maximum r value for r-shrink

n: the number of nodes

Table 17 Time limits in the experiments.

n	Time Limits (s)
20	10
50	20
100	100

n: the number of nodes

Table 18 Experimental results in solving 20-node instances.

Instance[a]	Optimum [21]	Proposed PSO				
		Excess	Found	Time(s)	Average	Std.dev.
p.20.00	407250.81	-	30/30	0.23	407250.81	0.00
p.20.01	446905.52	-	30/30	0.02	446905.52	0.00
p.20.02	335102.42	-	30/30	0.02	335102.42	0.00
p.20.03	488344.90	-	30/30	0.02	488344.90	0.00
p.20.04	516117.75	-	30/30	0.03	516117.75	0.00
p.20.05	300869.14	-	30/30	0.03	300869.14	0.00
p.20.06	250553.15	-	30/30	0.01	250553.15	0.00
p.20.07	347454.08	-	30/30	0.02	347454.08	0.00
p.20.08	390795.34	-	30/30	0.02	390795.34	0.00
p.20.09	447659.11	-	30/30	0.03	447659.11	0.00
p.20.10	316734.39	-	30/30	0.07	316734.39	0.00
p.20.11	289200.92	-	30/30	0.55	289200.92	0.00
p.20.12	314511.98	-	30/30	0.02	314511.98	0.00
p.20.13	346234.51	-	30/30	0.05	346234.51	0.00
p.20.14	301426.68	-	30/30	0.02	301426.68	0.00
p.20.15	457467.93	-	30/30	0.02	457467.93	0.00
p.20.16	484437.68	-	30/30	0.02	484437.68	0.00
p.20.17	380175.41	-	30/30	0.02	380175.41	0.00
p.20.18	320300.23	-	30/30	0.10	320300.23	0.00
p.20.19	461267.52	-	30/30	0.04	461267.52	0.00
p.20.20	403582.74	-	30/30	0.04	403582.74	0.00
p.20.21	271958.28	-	30/30	0.01	271958.28	0.00
p.20.22	328659.78	-	30/30	0.02	328659.78	0.00
p.20.23	326654.08	-	30/30	0.04	326654.08	0.00
p.20.24	395859.67	-	30/30	0.03	395859.67	0.00
p.20.25	453517.28	-	30/30	0.03	453517.28	0.00
p.20.26	461547.18	-	30/30	0.01	461547.18	0.00
p.20.27	389057.00	-	30/30	0.02	389057.00	0.00
p.20.28	279251.95	-	30/30	0.01	279251.95	0.00
p.20.29	299586.76	-	30/30	0.02	299586.76	0.00
Average		0.00%	30/30	0.05	373749.47	0.00

a. The 20-node instances can be obtained at http://dag.informatik.uni-kl.de/research/meb/ .

Table 19 Experimental results in solving 50-node instances.

Instance[a]	Optimum [21]	ELS [27]			HGA [28]			ACO [29]					Proposed PSO				
		Excess	Found	Time(s)	Excess	Found	Time(s)	Excess	Found	Time(s)	Average	Std.dev.	Excess	Found	Time(s)[b]	Average	Std.dev.
p50.00	399074.64	0.41%	15/30	57	0.88%	6/30	12.82	–	30/30	0.52	399074.64	0.00	–	30/30	0.81	399074.64	0.00
p50.01	373565.15	0.16%	5/30	47	0.36%	9/30	20.18	–	30/30	1.97	373565.15	0.00	–	30/30	6.75	373565.15	0.00
p50.02	393641.09	0.28%	13/30	46	–	30/30	9.76	–	30/30	2.84	393641.09	0.00	–	30/30	2.23	393641.09	0.00
p50.03	316801.09	1.71%	11/30	57	–	30/30	15.28	–	30/30	0.36	316801.09	0.00	–	30/30	0.96	316801.09	0.00
p50.04	325774.22	0.30%	25/30	40	0.40%	8/30	13.15	–	30/30	1.89	325774.22	0.00	–	30/30	1.31	325774.22	0.00
p50.05	382235.90	0.83%	16/30	31	–	30/30	9.61	–	30/30	3.42	382235.90	0.00	–	30/30	1.11	382235.90	0.00
p50.06	384438.46	–	30/30	29	–	30/30	9.20	–	30/30	3.54	384438.46	0.00	–	30/30	1.63	384438.46	0.00
p50.07	401836.85	0.54%	24/30	64	1.46%	1/30	13.24	–	30/30	0.54	401836.85	0.00	–	30/30	0.79	401836.85	0.00
p50.08	334418.45	–	30/30	29	–	30/30	6.71	–	30/30	0.73	334418.45	0.00	–	30/30	1.26	334418.45	0.00
p50.09	346732.05	3.29%	0/30	102	1.24%	17/30	21.84	–	30/30	1.88	346732.05	0.00	–	30/30	1.65	346732.05	0.00
p50.10	416783.45	1.16%	13/30	40	–	30/30	14.14	–	30/30	0.85	416783.45	0.00	–	30/30	1.50	416783.45	0.00
p50.11	369869.41	2.87%	1/30	28	0.32%	25/30	15.05	–	30/30	4.47	369869.41	0.00	–	30/30	1.22	369869.41	0.00
p50.12	392326.01	0.57%	7/30	66	0.90%	20/30	18.57	–	30/30	0.42	392326.01	0.00	–	30/30	0.64	392326.01	0.00
p50.13	400563.83	0.04%	29/30	74	0.09%	11/30	25.55	–	30/30	1.38	400563.83	0.00	–	30/30	1.69	400563.83	0.00
p50.14	388714.91	0.34%	3/30	11	–	30/30	4.12	0.00%	29/30	2.26	388763.13	264.14	0.00%	28/30	2.00	390532.51	6917.09
p50.15	371694.65	0.20%	5/30	35	–	30/30	13.09	–	30/30	1.12	371694.65	0.00	–	30/30	0.86	371694.65	0.00
p50.16	414587.42	0.30%	26/30	81	1.24%	1/30	30.19	–	30/30	0.48	414587.42	0.00	–	30/30	1.61	414587.42	0.00
p50.17	355937.07	1.88%	17/30	33	0.03%	28/30	14.37	–	30/30	0.38	355937.07	0.00	–	30/30	1.19	355937.07	0.00
p50.18	376617.33	0.24%	8/30	65	0.00%	24/30	16.17	–	30/30	0.55	376617.33	0.00	0.00%	29/30	1.00	376619.01	9.19
p50.19	335059.72	–	30/30	28	–	30/30	10.57	–	30/30	0.39	335059.72	0.00	–	30/30	0.56	335059.72	0.00
p50.20	414768.96	0.15%	0/30	35	0.13%	4/30	10.29	0.00%	21/30	5.85	414952.36	284.94	–	30/30	0.95	414768.96	0.00
p50.21	361354.27	–	30/30	41	–	30/30	10.61	–	30/30	0.29	361354.27	0.00	–	30/30	1.25	361354.27	0.00
p50.22	329043.51	–	30/30	14	–	30/30	8.32	–	30/30	0.74	329043.51	0.00	–	30/30	0.77	329043.51	0.00
p50.23	383321.04	–	30/30	109	–	30/30	12.49	–	30/30	1.06	383321.04	0.00	–	30/30	2.99	383321.04	0.00
p50.24	404855.92	0.07%	17/30	37	–	30/30	12.82	–	30/30	0.53	404855.92	0.00	–	30/30	0.87	404855.92	0.00
p50.25	363200.32	–	30/30	7	–	30/30	6.37	–	30/30	0.45	363200.32	0.00	–	30/30	0.28	363200.32	0.00
p50.26	406631.51	2.17%	2/30	60	0.30%	27/30	15.26	–	30/30	2.16	406631.51	0.00	–	30/30	1.77	406631.51	0.00
p50.27	451059.62	0.18%	22/30	40	0.01%	29/30	9.85	–	30/30	1.50	451059.62	0.00	–	30/30	1.40	451059.62	0.00
p50.28	415832.44	0.47%	23/30	78	0.16%	27/30	23.69	–	30/30	0.38	415832.44	0.00	–	30/30	0.91	415832.44	0.00
p50.29	380492.77	0.08%	27/30	18	–	30/30	8.10	–	30/30	1.92	380492.77	0.00	–	30/30	1.61	380492.77	0.00
Average		0.61%	17.3/30	46	0.25%	22.9/30	13.71	0.00%	29.7/30	1.51	379715.46	18.30	0.00%	29.9/30	1.45	379768.38	230.88

a. The 50-node instances can be obtained at http://dag.informatik.uni-kl.de/research/meb/ .

b. The computation time refers to the average time required to find the optimal solutions, and the trials which fail to find the optimal solutions are not included.

Table 20 Experimental results in solving 100-node instances.

Instance[a]	Best Known	ACO [29]				Proposed PSO					t-test[c]
		Excess	Average	Std.dev.	Time(s)	Excess	Average	Std.dev.	Found	Time(s)	
p100.00	340869.27	0.01%	340909.39	122.41	26.50	–	**340869.27**[b]	0.00	30/30	20.38	–
p100.01	355284.77	0.09%	355619.98	256.40	38.98	0.01%	**355320.76**	129.60	27/30	41.75	▲
p100.02	377145.59	–	377145.59	0.00	6.80	–	377145.59	0.00	30/30	11.69	–
p100.03	356942.53	0.09%	357246.73	186.58	23.78	0.09%	357246.75	149.47	1/30	57.31	–
p100.04	384446.36	0.09%	384781.20	170.28	21.10	–	**384446.36**	0.00	30/30	36.99	▲
p100.05	416758.58	–	416758.58	0.00	19.21	–	416758.58	0.00	30/30	30.38	–
p100.06	376408.49	0.76%	379266.65	1603.74	34.61	–	**376408.49**	0.00	30/30	19.21	▲
p100.07	343798.46	–	343798.46	0.00	10.95	0.11%	344181.03	1455.91	28/30	31.63	–
p100.08	372254.06	0.44%	373888.96	743.65	31.35	0.09%	**372594.12**	458.91	19/30	62.69	▲
p100.09	366993.89	–	366993.89	0.00	10.38	–	366993.89	0.00	30/30	23.50	–
p100.10	334579.00	–	334579.00	0.00	3.80	–	334579.00	0.00	30/30	17.35	–
p100.11	356219.14	–	356219.14	0.00	12.28	–	356219.14	0.00	30/30	36.23	–
p100.12	393854.17	0.11%	394305.00	618.38	33.81	0.06%	**394095.59**	357.55	6/30	43.16	–
p100.13	331270.37	–	331270.37	0.00	11.89	0.07%	331515.93	777.92	27/30	48.89	–
p100.14	344175.57	0.21%	344883.10	1013.85	29.09	–	**344175.57**	0.00	30/30	27.07	▲
p100.15	352884.55	0.01%	352930.55	76.34	40.42	0.00%	**352897.81**	34.38	26/30	49.88	▲
p100.16	338713.69	0.37%	339968.85	1007.75	43.65	0.06%	**338925.05**	540.78	20/30	55.00	▲
p100.17	374059.25	1.11%	378223.85	3362.87	41.84	0.40%	**375549.92**	1611.15	15/30	46.13	▲
p100.18	331926.13	1.56%	337088.24	2625.18	23.70	–	**331926.13**	0.00	30/30	37.47	▲
p100.19	365078.37	–	365078.37	0.00	13.18	–	365078.37	0.00	30/30	12.23	–
p100.20	355078.27	0.22%	355874.33	1538.21	46.43	0.53%	356963.57	1928.98	13/30	40.89	▽
p100.21	362204.29	0.01%	362251.54	179.82	22.39	0.02%	362267.30	271.90	28/30	59.59	–
p100.22	366125.96	–	366125.96	0.00	39.12	0.01%	366146.09	90.65	27/30	51.98	–
p100.23	409062.55	0.13%	409614.57	1144.29	42.10	0.04%	**409228.64**	309.21	22/30	59.27	–
p100.24	357772.11	0.24%	358616.47	827.94	36.93	0.05%	**357955.48**	418.80	25/30	57.70	▲
p100.25	357191.63	0.27%	358138.87	708.06	54.25	0.06%	**357396.28**	357.01	16/30	58.17	▲
p100.26	352148.02	–	352148.02	0.00	28.37	0.06%	352366.35	667.84	27/30	36.56	–
p100.27	370033.07	0.32%	371208.73	991.17	47.63	0.21%	**370820.22**	790.17	13/30	60.49	–
p100.28	348889.36	0.20%	349602.54	1085.17	35.84	–	**348889.36**	0.00	30/30	16.51	▲
p100.29	357595.04	0.07%	357862.43	420.10	30.67	0.16%	358165.04	487.32	10/30	62.19	▽
Average		0.21%	362413.31	622.74	28.70	0.07%	**361904.19**	361.25	23.67/30	40.41	

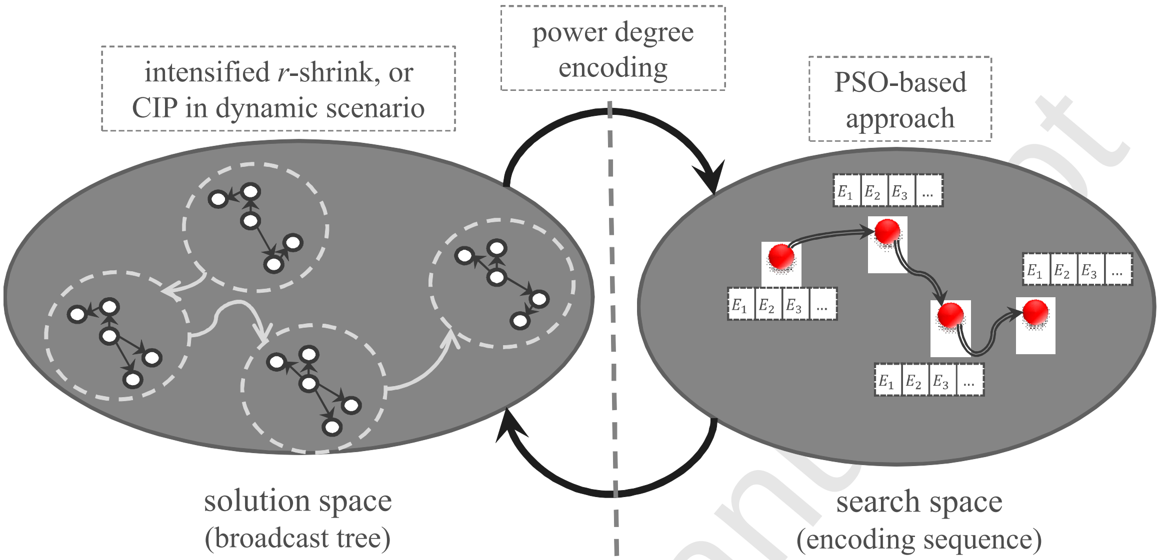
a. The 100-node instances can be obtained at http://www.lsi.upc.edu/~hhernandez/mem.
b. The data in the bold represent our proposed PSO outperforms ACO in terms of average solution quality.
c. '▲': PSO is significantly better than ACO. '▽': PSO is significantly worse than ACO. '–': Not statistically significant.

Table 21 Experimental results of intensified *r*-shrink.

Instance	Best Known	BIP + *r*-shrink			BIP + intensified *r*-shrink			PSO + *r*-shrink		PSO + intensified *r*-shrink	
		Excess	Obj. value	Time(s)	Excess	Obj. value	Time(s)	Excess	Obj. value	Excess	Obj. value
p100.00	340869.27	15.26%	392882.19	0.002	15.26%	392882.19	0.003	–	340869.27	–	340869.27
p100.01	355284.77	8.65%	386024.28	0.002	8.65%	386024.28	0.004	0.00%	355301.81	0.01%	355320.76
p100.02	377145.59	11.77%	421535.77	0.003	11.28%	**419694.61**[a]	0.003	0.60%	379394.96	–	**377145.59**[a]
p100.03	356942.53	6.62%	380582.22	0.002	6.62%	380582.22	0.003	0.19%	357620.99	0.09%	**357246.75**
p100.04	384446.36	9.70%	421756.05	0.002	9.61%	**421394.03**	0.002	0.98%	388210.08	–	**384446.36**
p100.05	416758.58	10.66%	461204.71	0.002	10.66%	461204.71	0.002	0.28%	417923.89	–	**416758.58**
p100.06	376408.49	8.54%	408560.85	0.002	8.54%	408560.85	0.004	0.04%	376557.14	–	**376408.49**
p100.07	343798.46	16.88%	401831.14	0.002	16.88%	401831.14	0.003	0.30%	344838.17	0.11%	**344181.03**
p100.08	372254.06	11.41%	414736.54	0.003	11.05%	**413375.82**	0.003	1.04%	376136.56	0.09%	**372594.12**
p100.09	366993.89	18.69%	435568.18	0.002	18.13%	**433518.49**	0.004	0.02%	367059.27	–	**366993.89**
p100.10	334579.00	13.88%	381011.32	0.002	13.88%	381011.32	0.002	0.13%	335000.49	–	**334579.00**
p100.11	356219.14	11.81%	398282.08	0.002	11.51%	**397222.37**	0.002	0.21%	356958.46	–	**356219.14**
p100.12	393854.17	10.71%	436048.36	0.002	10.71%	436048.36	0.002	1.06%	398045.37	0.06%	**394095.59**
p100.13	331270.37	20.56%	399381.12	0.001	20.56%	399381.12	0.003	0.33%	332351.21	0.07%	**331515.93**
p100.14	344175.57	9.26%	376032.03	0.002	8.67%	**374006.99**	0.003	0.03%	344273.32	–	**344175.57**
p100.15	352884.55	7.42%	379065.52	0.002	7.42%	379065.52	0.002	0.02%	352957.07	0.00%	**352897.81**
p100.16	338713.69	10.09%	372898.29	0.002	10.09%	372898.29	0.003	0.55%	340588.63	0.06%	**338925.05**
p100.17	374059.25	8.66%	406460.51	0.002	8.30%	**405098.58**	0.002	1.00%	377815.44	0.40%	**375549.92**
p100.18	331926.13	10.96%	368321.6	0.002	7.41%	**356526.16**	0.004	0.71%	334270.12	–	**331926.13**
p100.19	365078.37	17.98%	430701.95	0.002	17.75%	**429884.32**	0.004	–	365078.37	–	365078.37
p100.20	355078.27	9.06%	387253.06	0.002	9.06%	387253.06	0.002	1.09%	358942.54	0.53%	**356963.57**
p100.21	362204.29	11.53%	403978.35	0.002	10.52%	**400308.56**	0.002	0.16%	362795.60	0.02%	**362267.30**
p100.22	366125.96	13.44%	415337.68	0.002	13.44%	415337.68	0.003	0.69%	368661.24	0.01%	**366146.09**
p100.23	409062.55	11.18%	454805.00	0.002	11.10%	**454454.01**	0.003	0.57%	411407.28	0.04%	**409228.64**
p100.24	357772.11	8.79%	389233.09	0.002	8.79%	389233.09	0.003	0.66%	360129.66	0.05%	**357955.48**
p100.25	357191.63	8.94%	389124.06	0.002	6.45%	**380240.03**	0.002	0.48%	358905.07	0.06%	**357396.28**
p100.26	352148.02	18.11%	415937.61	0.002	17.64%	**414277.11**	0.002	0.16%	352725.13	0.06%	**352366.35**
p100.27	370033.07	9.69%	405906.76	0.002	9.69%	405906.76	0.003	0.72%	372699.00	0.21%	**370820.22**
p100.28	348889.36	8.55%	378727.93	0.002	8.48%	**378482.93**	0.004	0.19%	349547.80	–	**348889.36**
p100.29	357595.04	16.63%	417051.09	0.003	16.57%	**416836.71**	0.004	0.36%	358894.08	0.16%	**358165.04**
Average		11.85%	404341.31	0.00207	11.49%	**403084.71**	0.00287	0.42%	363198.60	0.07%	**361904.19**

a. The data in bold represents BIP + intensified *r*-shrink outperforms BIP + *r*-shrink or PSO + intensified *r*-shrink outperforms PSO + *r*-shrink.

Graphical Abstract



Research Highlights

- A PSO-based approach for the static minimum energy broadcast (MEB) problem
- A new encoding scheme to transform solutions between solution and search space
- The intensified *r*-shrink based on the analysis of depth information
- An efficient and effective heuristic for the dynamic MEB problem

Table 11 Experimental results in solving 100-node instances by simple heuristics.

Instance	Best Solution	**ABC** [17]		**BIP** [14]		**GPBE** [15]		**MST** [14]		**SPT** [14]		**Proposed CIP**	
		Dev.Pct.[a]	Obj, value	Dev.Pct.	Obj. value	Dev.Pct.	Obj. value	Dev.Pct.	Obj. value	Dev.Pct.	Obj. value	Dev.Pct.	Obj. value
p100.00	445228.70	0.00	445228.70	0.01	450275.55	0.33	590017.43	0.07	478402.14	0.19	529976.08	0.01	449511.30
p100.01	421678.40	0.04	440516.38	0.03	434920.86	0.07	449112.18	0.07	452666.88	0.35	568452.81	0.00	**421678.40**[b]
p100.02	447953.30	0.04	466255.41	0.04	467185.46	0.14	510375.36	0.16	521601.94	0.34	602434.70	0.00	**447953.30**
p100.03	408080.42	0.06	431417.64	0.09	445615.58	0.05	429246.20	0.12	457287.91	0.53	624323.76	0.00	**408080.42**
p100.04	465386.88	0.02	472400.87	0.00	465386.88	0.02	473604.74	0.07	496650.90	0.29	600454.11	0.01	470922.00
p100.05	511466.39	0.02	520386.55	0.01	517040.38	0.08	553924.96	0.05	538098.70	0.31	668836.65	0.00	**511466.39**
p100.06	460143.42	0.10	504217.43	0.04	478615.38	0.00	460143.42	0.12	517507.12	0.20	552986.15	0.00	461433.50
p100.07	450138.55	0.03	461623.02	0.00	450138.55	0.01	454798.19	0.01	455401.78	0.41	635856.03	0.01	452874.78
p100.08	466767.17	0.03	480034.73	0.00	468277.72	0.13	528756.02	0.07	497475.62	0.33	621286.44	0.00	**466767.17**
p100.09	466000.16	0.01	469465.51	0.01	468861.19	0.00	466000.16	0.11	515282.56	0.47	683606.73	0.00	466591.59
p100.10	387623.01	0.08	417968.21	0.05	407806.54	0.00	387623.01	0.14	443232.02	0.46	567558.40	0.03	397389.62
p100.11	425755.20	0.07	454912.37	0.07	456482.54	0.10	468356.15	0.10	468159.22	0.34	572222.13	0.00	**425755.20**
p100.12	477202.40	0.04	494307.81	0.02	487388.96	0.17	556089.54	0.05	500999.26	0.33	636276.68	0.00	**477202.40**
p100.13	410825.11	0.05	432388.89	0.06	434600.54	0.00	410825.11	0.14	469892.83	0.27	521042.44	0.05	430316.55
p100.14	409127.08	0.00	410879.55	0.03	419692.94	0.00	409127.08	0.10	451288.42	0.41	578888.48	0.02	416522.45
p100.15	415269.77	0.00	415269.77	0.00	416404.90	0.10	454985.62	0.09	451008.48	0.31	542448.10	0.01	418129.28
p100.16	395943.77	0.06	418976.72	0.04	413254.91	0.00	395943.77	0.08	427486.11	0.37	542676.26	0.04	411513.03
p100.17	432518.10	0.05	454946.06	0.02	442166.58	0.02	440825.98	0.17	505088.13	0.39	602841.68	0.00	**432518.10**
p100.18	390968.46	0.02	398877.33	0.02	397526.80	0.07	417672.59	0.12	437840.16	0.39	544627.94	0.00	**390968.46**
p100.19	442033.79	0.06	466669.16	0.06	468763.42	0.14	502863.21	0.14	504598.48	0.21	534170.30	0.00	**442033.79**
p100.20	436625.21	0.00	436625.21	0.02	444943.92	0.26	548580.73	0.10	480030.24	0.36	594471.80	0.02	443405.97
p100.21	416587.71	0.10	459744.70	0.06	443403.83	0.00	416587.71	0.14	473167.51	0.54	639506.55	0.04	434475.57
p100.22	429210.03	0.08	463833.01	0.03	443943.48	0.16	497791.85	0.07	457863.09	0.27	544960.23	0.00	**429210.03**
p100.23	487127.12	0.07	520934.69	0.08	523716.05	0.00	487127.12	0.16	563377.27	0.35	659512.23	0.01	491176.21
p100.24	431005.17	0.02	441738.50	0.00	431005.17	0.06	455816.99	0.10	473933.98	0.36	584600.46	0.01	433758.69
p100.25	429398.49	0.12	481759.36	0.02	436269.59	0.22	524631.48	0.13	484813.40	0.40	602578.32	0.00	**429398.49**
p100.26	403960.95	0.19	482240.29	0.13	454831.86	0.00	403960.95	0.28	515650.84	0.58	638640.95	0.12	452914.72
p100.27	434256.03	0.01	438344.91	0.02	444542.65	0.14	493428.13	0.06	461251.20	0.26	547609.17	0.00	**434256.03**
p100.28	423277.51	0.05	443415.17	0.06	448086.30	0.00	423277.51	0.16	490883.66	0.62	686114.56	0.02	433323.49
p100.29	423624.42	0.08	458286.40	0.08	459516.13	0.00	423624.42	0.11	468384.69	0.39	587431.36	0.07	454117.88
Average		0.05	456122.15	0.04	450688.82	0.08	467837.25	0.11	481977.48	0.37	593879.72	0.01	**441188.83**

a. Deviation percentage from the best solution among algorithms.

b. The data in bold represent our proposed CIP performs the best among algorithms.