Energy-aware Topology Control for Wireless Sensor Networks Using Memetic Algorithms

Andreas Konstantinidis [a,b] Kun Yang [a,*] Hsiao-Hwa Chen [c] Qingfu Zhang [b]

[a] *Department Of Electronic Systems Engineering, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK*
[b] *Department Of Computer Science, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK*
[c] *Institute of Communications Engineering, National Sun Yat - Sen University, Taiwan*

Abstract

Cost-effective topology control is critical in wireless sensor networks. While much research has been carried out in this aspect using various methods, no attention has been made on utilizing modern heuristics for this purpose. This paper proposes a memetic algorithm-based solution for energy-aware topology control for wireless sensor networks. This algorithm (called ToCMA), using a combination of problem-specific light-weighted local search and genetic algorithm, is able to solve the minimum energy network connectivity (MENC) this NP-hard problem in an approximated manner that performs better than the classical minimum spanning tree (MST) solution. The outcomes of ToCMA can also be utilized for various network optimization and fault-tolerant purposes.

Key words: wireless sensor networks, topology control, energy awareness, memetic algorithms, genetic algorithms

1. Introduction

Wireless sensor networks have attracted a phenomenon of research recently due to its ability of collecting data in hostile environment and reporting it back to a sink [1]. This ability brings great impact on different walks of people's life. Wireless sensor networks face several challenges with the most significant one being the energy consumption. Many protocols have been proposed to reduce the power consumption of sensors to keep the lifespan of a sensor network as long as possible because usually recharging sensors' battery is not easy. An issue that is highly related to power consumption of a wireless sensor network is network topology. Unlike wired networks where the link topology is fixed after the networks are deployed, wireless sensor networks is difficult to maintain such a feature because of the randomness of sensor node distribution at the time of deployment and the transmission power of each node. The multi-hop nature of the wireless sensor network routing renders the unevenness of the power distribution across sensor nodes in the network concerned, which results in gradual topology change of the whole network. For instance, the previously connected network might become partitioned due to battery exhaust of a critical connecting node. If the transmission power of each sensor is properly assigned and dynamically adjusted then a message from any node should be able to be routed to any other node in the network. In this case the network is considered as connected. Connectivity is a basic requirement for sensor networks. Topology control assures the basic essential connectivity of a wire-

* Corresponding author. Tel: +44 (0) 1206 872449 Fax: +44 (0) 1206 872900

Email addresses: akonst@essex.ac.uk (Andreas Konstantinidis), kunyang@essex.ac.uk (Kun Yang), hshwchen@ieee.org (Hsiao-Hwa Chen), qzhang@essex.ac.uk (Qingfu Zhang).

less sensor network and also reduces the number of nodes participating in forwarding and routing packets without diminishing coverage [2]. Moreover an efficient deployment of sensors usually offers redundancy in sensor networks coverage [2].

The wireless sensor network connectivity problem has been intensively researched using various graph or set theories, with the most commonly used tool being connected dominating set [3]. While wireless sensor networks share many similarities with wireless mobile ad hoc networks (MANET), the energy consumption is more a concern in sensor networks than it is in MANETs. Much research has been carried out on the energy-awareness of sensor networks, especially from the perspective of energy-efficient routing [4–6] whose focus is to find a most energy-efficient route given the current energy status of each node in the networks with one objective being prolonging the network's lifespan. The problem of minimizing the transmission power of each node in the network, which results in minimizing the energy consumption of the network, while keeping its global connectivity at the same time, is termed as the minimum energy network connectivity problem (MENC) by [7]. It has been proved that MENC is a NP-complete problem [7]. Network connectivity problem sometimes is called topology control.

Several heuristics have been developed to solve the MENC problem [7–9]. X. Cheng et al. [8] propose two heuristics based on a Minimum Spanning Tree (MST) [10] and a Broadcast Incremental Power (BIP [11]) method respectively. Based on the above work, M. Cheng et al. [7] present further improvement to the work in [8], e.g., a minimum incremental power (MIP) tree algorithm is designed. A. K. Das [9] uses a heuristic called r-shrink algorithm which again is based on BIP. BIP is a greedy heuristic proposed by Wieselthier et al. [11]. BIP takes advantage of the broadcast nature of wireless transmissions and computes a broadcast tree. This is actualized by adding nodes once at a time and at each step the less expensive action is selected. To the best of our knowledge, no research work has utilized metaheuristic, for example Genetic algorithms (GA) [12,13] or Memetic Algorithms (MA) [14–17] which combines GA with Local Search (LS). This combination has been proved very successful in dealing with hard and complex problems [14,15,18–20]. This paper is to investigate how MA can be applied to a hard problem in wireless sensor networks. In particular, we select the MENC problem and research on how MA is used to obtain performance gain. MAs [14], as inspired by the cultural evolution, employ one or more problem-specific heuristics to improve and/or repair the solutions generated by the genetic algorithm's operators (i.e. crossover and mutation).

The ToCMA (Topology Control using Memetic Algorithm) algorithm proposed in this paper generates many different solutions and explores in an effective manner the solution space by using searching and genetic algorithm operators. Then ToCMA employs different problem enriched algorithms to maintain the global connectivity of the network. It checks if the network is strongly connected and if it is not then ToCMA repairs it. Furthermore, ToCMA employs an improvement procedure to further minimize the overall energy consumption of the network as much as possible. Considering the resource constrained nature of sensors and the complexity of heuristic algorithms, ToCMA is designed to run offline and usually on the sink of a sensor network. The calculated solutions are distributed to the sensors in a network by the sink node using either multiple hop broadcasting or direct-communicating broadcasting.

The rest of the paper is organized as follows. In Section 2 we describe the network assumptions, the MENC problem itself and a briefing on memetic algorithm. Section 3 details the proposed algorithm step by step. Section 4 presents the performance evaluation of ToCMA against MST. Finally the paper concludes in Section 5.

The main contribution of the paper lies in threefold. Firstly, an MA-based new methodology is proposed for solving energy-aware topology control problem in wireless sensor networks; secondly, a way of encoding energy information for each node into a chromosome is proposed, and finally the proposed ToCMA has demonstrated its out-performance over the existing MST solution, amongst other network benefits.

2. Preliminaries

2.1. *Network Assumptions*

Since our work is largely inspired by M. Cheng et al. [7] and indeed is to solve the MENC problem defined by [7], the network model and energy consumption follow the similar assumptions as these in [7]. We assume the wireless sensor networks investigated have the following features:

– The sensors in the network are stationary and lo-

Table 1
Notations

Notation	Meaning
n	total number of sensor nodes in the sensor network concerned.
p_i	power assigned to node. i
Δ	the maximum power a node can be assigned to.
α	path loss exponent $2 \preceq \alpha \preceq 4$.
Pop_t	the population of generation t the population considered as the current population of generation t. Pop_0 stands for the initial population. A population is consisted of a number of chromosomes (or individuals).
μ	total number of chromosomes (or individuals) in a population.
c_i	chromosome i of a population with the index i indicating the position of the chromosome in the population concerned.
η	the maximum number of generations(ToCMA termination criterion).
$BestInd$	set that stores in an increasing order the best μ individuals of all generations.
Pop_t^{IF}	the set of infeasible solutions of population t.
Pop_t^{F}	the set of feasible solutions in population t.
Pop_t^{REP}	the set of μ feasible solutions that have been repaired.
Pop_t^{IMP}	the set of μ feasible solutions that have been improved.
Pop_{t+1}^{Crs}	a new population $t+1$ generated after crossover.
Pop_{t+1}^{Mut}	a new population $t+1$ generated after mutation.

cated in a two-dimensional plane. It is also assumed that the location of each sensor node can be obtained after the deployment, by using any positioning technology. The location information will be used for calculating the distance between two sensor nodes.

- Omnidirectional antenna [21] is used for each sensor. This means that a sensor radiates and receives equally in all directions. If a sensor transmits with a power level:

$$p_t = \zeta \times d^{\alpha} \tag{1}$$

then any sensor within the distance d and a power threshold ζ can receive the signal. The path loss exponent α is between 2 and 4 [7]. Suppose there are two nodes n_i and n_j then the distance between these two nodes can be calculated by using the Euclidean distance formula, $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, where (x_i, y_i) and (x_j, y_j) are the coordinates of nodes n_i and n_j respectively. The power threshold ζ is considered a constant and it is ignored since the receivers in the network have the same power threshold.
- ToCMA uses transmission power in energy calculation without considering the transmission time. The same assumption is adopted by [7].
- Sensor nodes can operate in different initial power levels, with a lower and an upper bound. This consequently leads to asymmetric wireless links and a directed graph. The asymmetry of the communication links combined with a request for a different initial power level makes complex the problem and renders the topology control problem more challenging. Note that only symmetric links are considered in [8].

2.2. *Problem Definition*

In a wireless network, if for each node, there is a route to reach any other node in the same network, then such a network is regarded *strongly connected* [7]. We use the same notations as that in [7] to describe the energy-aware topology control problem as follows. Let V denote the set of wireless sensor nodes and $G(V, E)$ denote the super-graph on V that contains all possible edges if each node transmits at its maximum transmission power. The edge set E of G is constructed in such a manner that there is a directed edge from u to v if and only if u can reach v using its maximum transmission power. Graph G

sets an upper bound on the maximum connectivity that a wireless network can have. The topology control algorithm returns a topology T constructed from G, i.e., T is a subgraph of G on V. A wireless sensor network should fulfil the following connectivity requirement: for any pair of nodes u and v, if there is a path from u to v in G then there is also a path from u to v in T.

The formal definition of the MENC problem is given as follows [7]:

Given a set of wireless nodes $V = n_1, n_2, ..., n_n$ *and the cost function* $F : (V, V) \rightarrow Z$*, MENC is to determine a power assignment of nodes* $P : V \rightarrow Z$ *such that:*

(i) The induced directed graph T *is strongly connected.*

(ii) The total energy consumption of the network $\sum_{i=1}^{n} p_i$ *is minimized where* p_i *denotes the power assigned to node* n_i *and is calculated via Formula (1).*

2.3. *Memetic Algorithm*

A memetic algorithm (MA) is a combination of a genetic algorithm and a local search [14] and it is an important optimization approach with successes in a variety of classical NP-hard optimization problems [14–17]. It is based on the principle of evolution operations such as crossover and mutation and the concept of fitness. It utilizes various problem specific heuristics to improve and/or repair the solutions generated by evolution.

In MA each *solution* is usually encoded as an integer string, with each integer representing different physical parameter that is specific to the problem to be solved. A solution is termed as an *individual* in a *population*. A population is associated with certain *generation t* in the whole evolution of individuals. The initial populations are usually generated in a random or controlled manner and then the evolution of these populations are carried out by the genetic operators such as crossover and mutation. Local search is utilized to check the feasibility of each population, a result of which is that the whole solutions in a generation are divided into two groups: one comprising the feasible solutions (denoted as Pop_t^F) and the other comprising the infeasible solutions(denoted as Pop_t^{IF}). Then a repair procedure is invoked on Pop_t^{IF} trying to "repair" them to feasible solutions. Even the feasible solutions can be further improved to obtain a better fitness. Here *fitness* is used to express how good a solution is, i.e., how close it is to the optimal solution. The proper definition of a fitness function is as important as expressing the solution of a problem into an integer string. A well-defined MA should be able to converge, i.e., the newer solutions are closer to the optimal one. The MA algorithm proposed in this paper for the MENC problem, ToCMA, follows the above procedure, which is also illustrated in Figure 1.

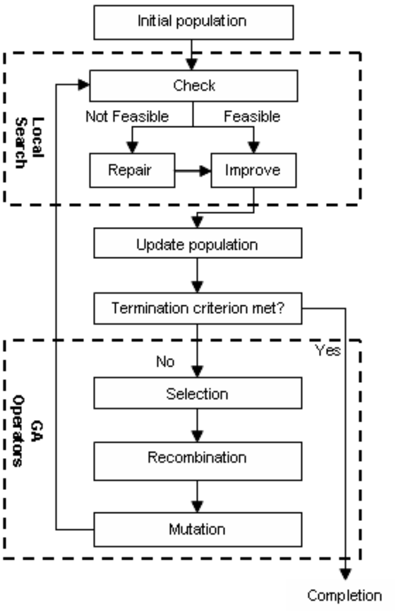


Fig. 1. ToCMA Flowchart

Table 1 summarizes a list of the main notations and their meanings to be used in the rest of the paper. Note that the three terms, individual, chromosome, and solution, represent the same meaning but from different points of view. For instance, a solution to a problem is represented as an individual, and from genetics' perspective an individual is a chromosome which again is composed of multiple genes.

3. ToCMA Algorithm

This section starts with a presentation as to how the MENC problem is represented by ToCMA, search space and the fitness function of a solution. Then it gives a detailed presentation as to how each step is designed and implemented for the MENC problem following the ToCMA flow in Figure 1.

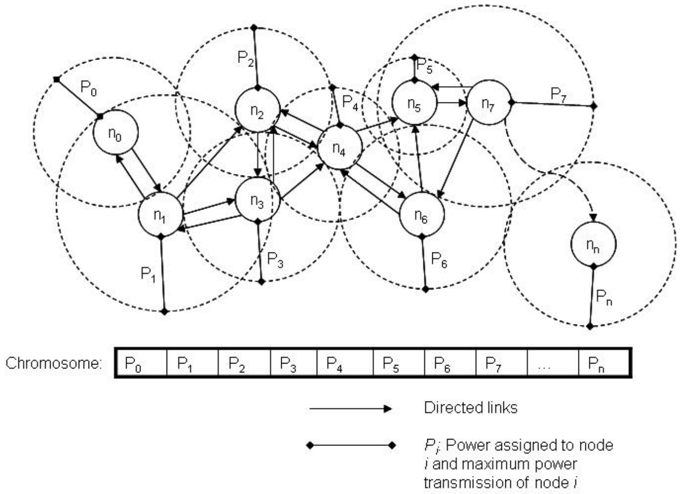

P0
P1
P2
P3
P4
P5
P6
P7
Pn
n0
n1
n2
n3
n4
n5
n6
n7
nn
Chromosome:
P0 P1 P2 P3 P4 P5 P6 P7 ... Pn
Directed links
Pi: Power assigned to node i and maximum power transmission of node i

Fig. 2. Example of power assignment and its encoding scheme

3.1. *Genetic Representation*

A ToCMA solution to the MENC problem is represented by a positive integer string where 1) the integer numbers of the string are the power levels assigned to each node respectively and 2) the position of an integer number in the string represents the node id. Here it is assumed that all sensor nodes in the sensor network concerned are numbered from 1 to n where n is the total number of nodes in the sensor network. From Section 2.1 we know that the sensors are stationary, i.e., the location of each sensor is fixed after deployment, then the assignment of power to the sensor nodes become the only factor that affects the connectivity of the sensor network (suppose interference is not considered as in [7]). In GA terminology [13], a solution is represented as a chromosome. So a solution or a chromosome is represented as $c_i = (p_{1,i}, p_{2,i}, ..., p_{n,i})$ where c_i denotes the i-th solution in the solution space (or population) and $p_{i,j}, j \in 1, 2, ..., n$ denotes the power assigned to the j-th sensor as far as the i-th solution is concerned. In general, the composing entities of a chromosome are called *genes*. In ToCMA, genes are the power of sensors. An example of chromosome (power assignment) encoding from node n_1 to n_n is depicted in Figure 2.

The whole solutions (or chromosomes)constitute the solution space (i.e., the *search space* [15])$SS = \{c_i | i \in \{1, 2, ..., \mu\}\}$. In ToCMA it is assumed that there are μ solutions in each generation. If we denote the whole population in generation t as Pop_t then we have $Pop_t = \{c_1, c_2, ..., c_\mu\}$. This search space is to be used by ToCMA to find a best solution in the current population.

3.2. *Fitness Function*

The quality of each solution is measured by a *fitness function* [22]. In ToCMA, the fitness function of a solution is defined as the sum of the power assigned to each gene (i.e., sensor node), namely:

$$f(c_i) = \sum_{j=1}^{n} p_{j,i} \qquad (2)$$

To the MENC problem, the smaller a solution's fitness value is the better the solution is. An optimal solution is defined as a solution that has minimal $f(c_i)$, namely, $f(c_{optimal}) = min(f(c_i), i = 1, 2, ..., \mu)$. The fitness function is to be used as a criterion when selecting a chromosome.

3.3. *Population Initialization*

ToCMA starts by creating an initial population. In general, there are two issues to be considered when carrying out population initialization of a genetic algorithm: the initial population size and the procedure to initialize the population [22]. Firstly, it is generally true that the bigger the population size is the more chance and faster that a good solution can be found. However, a large population demands more computation and memory. Secondly, there are two ways to generate the initial population: heuristic initialization and random initialization.

ToCMA adopts a random process to generate its initial populations due to the speediness and less complexity of this method. However, ToCMA can equally take benefit of more intelligent initialization method [23,24] but at a cost of more computational complexity. A random number generator is used by ToCMA to generate $p_i, 0 \leq p_i \leq \Delta$, for each sensor node i where Δ is the maximum power a node can be assigned to. The following steps detail the population initialization procedure of ToCMA.

```
|ToCMA Population Initialization|
Step 1 Parameter setting: Set $g$,$\Delta$,$n$ = the number of nodes in the network,$M$,$\mu$;
$Pop_0 = \emptyset$; $i = 0$.
Step 2: Randomly pick numbers from 0 to $\Delta$ and assign them to the $n$ nodes of the network. Create an integer string $c_i$ using the following procedure:
  for $j = 1$ to $n$ do
    get the power assigned to node $n_i$ and put it at the $j$-th position in the string $c_i$;
    $Pop_0 \leftarrow Pop_0 \cup c_i$;
    $i \leftarrow i + 1$;
Step 3: Repeat $\mu$ times the step 1 to generate the initial population $Pop_0$.
```

The computational time complexity of the population initialization is $O(\mu.n)$.

By this point, based on the location and the power of the nodes, a graph G_i can be created corresponding to each c_i. The above population initialization procedure does not guarantee the feasibility of each $G_i, i = 1, 2, ..., \mu$ in Pop_0. As such a checking procedure, as part of local search procedure, has to get involved.

3.4. *Local Search:Checking, Repairing and Improvement*

3.4.1. *Checking Function*

A checking process is to check if a solution is feasible. In terms of ToCMA, it is to check if the graph generated based on this solution gives a directed strongly connected network. Checking constitutes the first step of local search procedure. Based on the analysis of sensor network topology, the following four cases as depicted in Figure 3 might cause an infeasible network topology.

In Case #1, as shown in Figure 3(a), there is one or more *totally isolated node* in the network represented by graph g, for example, the node circled. This case can be formally expressed as: $\exists n_i \in V(g), inDegree(n_i) = 0 \wedge outDegree(n_i) = 0$ where $V(g)$ denotes the set of the nodes in graph g and $inDegree(n_i)$ and $outDegree(n_i)$ stands for the indegree and out-degree of node n_i respectively. Such a solution is put into the infeasible set of the current generation t, i.e.,
$Pop_t^{IF} \leftarrow c = \{g(V, E) | \exists n_i \in V, inDegree(n_i) = 0$
$\wedge outDegree(n_i) = 0\}$

In Case #2, as shown in Figure 3(b), there are one or more *one-way isolated groups of nodes* in the network represented by graph g, for example the three groups of nodes g_1,g_2,g_3 circled. The first group, g_1, is composed by n_0, n_1, n_2, n_3, n_4, the second group, g_2, by n_5, n_6, n_7, n_8 and the third group, g_3, by $n_9, n_{10}, n_{11}, n_{12}$. In this case one group can reach the other groups but it cannot be reached by them, while another one can be reached by the other groups but it cannot reach them. For example, in Figure 3(b), g_1 can reach g_3 but it cannot reach g_2, where at the same time g_1 is reachable by g_2 and it is not reachable by g_3. This case is formally expressed as:
1) these three graphs do not share any common node, i.e.,
$V(g_1) \in V(g) \wedge V(g_2) \in V(g) \wedge V(g_3) \in V(g) \wedge V(g_1) \cap V(g_2) \cap V(g_3) = 0$ (we denote this condition as C1); and
2) there is only one link connecting each sub-graph,
$(\bigcup_{x_1 \in V(g_1)} neighbour(x_1)) \cap (\bigcup_{y \in V(g_2)} neighbour(y)) = 1 \wedge (\bigcup_{x_2 \in V(g_1)} neighbour(x_2)) \cap (\bigcup_{k \in V(g_3)} neighbour(k)) = 1$ (we denote this condition as C2). Any solution falling into this Case is also put into the infeasible set of the current generation t, i.e.,
$Pop_t^{IF} \leftarrow c = \{g(V, E) | \exists g_1, g_2, g_3, C1 \wedge C2\}$ A special occasion of this case is when there is a one way isolated node instead of a group of nodes as it shown in Figure3(b) by nodes n_{13} and n_{14}.

In Case #3, as shown in Figure 3(c), there is one or more *loop* in the network represented by graph g. For example, there is a loop between node n_1 and node n_2. For directed graph, this means n_1 and n_2 are each other's only next-hop neighbouring node. This case is formally expressed as: $\exists n_i, n_j \in V(g), (n_j \in neighbour(n_i) \wedge outDegree(n_i) == 1) \wedge (n_i \in neighbour(n_j) \wedge outDegree(n_j) == 1)$ where $neighbour(n_i)$ means the one-hop neighbours of node n_i. Any solution falling into this Case is also put into the infeasible set of the current generation t, i.e.,
$Pop_t^{IF} \leftarrow c = \{g(V, E) | \exists n_i, n_j \in V, (n_j \in$

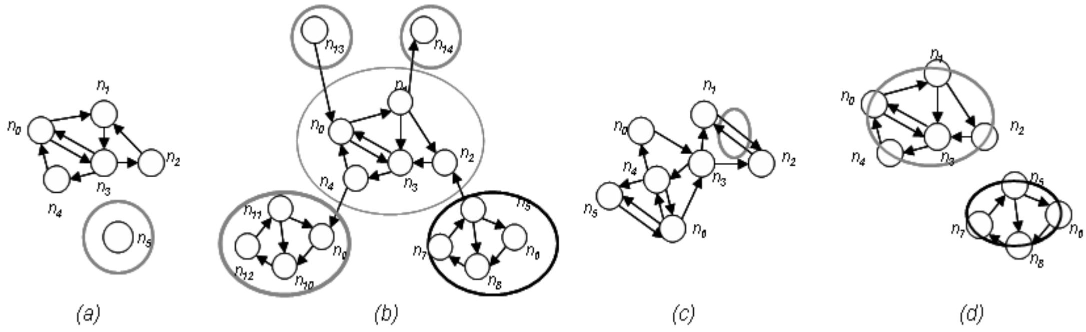


Fig. 3. Cases of Infeasible Solutions

$neighbour(n_i) \wedge outDegree(n_i) == 1)$
$\wedge(n_i \in neighbour(n_j) \wedge outDegree(n_j) == 1)\}$

In Case #4, as shown in Figure 3(d), *partition* occurs in the network. For example, there are two sub-networks in the graph, one composed of nodes n_0, n_1, n_2, n_3, n_4 and the other composed of node n_5, n_6, n_7, n_8 and there is no edge linking these two sub-networks. This case is formally expressed as: there are at least two sub-graph g_1, g_2 where the following conditions hold:
1) these two graphs do not share any common node, i.e.,
$V(g_1) \in V(g) \wedge V(g_2) \in V(g) \wedge V(g_1) \cap V(g_2) = \emptyset$ (we denote this condition as C1); and
2) there is no link connecting these two sub-graphs, i.e., they are not each other's neighbours:
$(\bigcup_{x\in V(g_1)} neighbour(x)) \cap (\bigcup_{y \in V(g_2)} neighbour(y)) = \emptyset$ (we denote this condition as C2).

Actually Case #1 is a special case of Case #4. Any solution falling into this Case is also put into the infeasible set of the current generation t, i.e., $Pop_t^{IF} \leftarrow c = \{g(V, E)|\exists g_1, g_2, C1 \wedge C2\}$

Any other solutions, which are feasible solutions, are put into the feasible population set:Pop_t^F . The checking function of ToCMA works as follows:

```
**Input:** a set of $\mu$ chromosomes each representing
a possible solution, i.e.,$Pop_t = c_1, c_2, ..., c_\mu$
**Output:** two sets of chromosomes: $Pop_t^{IF}$
representing a set of infeasible solutions and
$Pop_t^F$ representing a set of feasible solutions.
**for**$i = 1$ to $\mu$ **do**
    **if** $c_i \in Pop_t$ satisfies Case #1, or Case #2, or
    Case #3, or Case #4 **then**
        $Pop_t^{IF} \leftarrow c_i$;
    **else**
        $Pop_t^F \leftarrow c_i$;
```

All infeasible solutions are to be forwarded to a repairing process for repair and the feasible solutions go directly to the improvement process.

The computational time complexity of the checking algorithm is $O(\mu.n^2)$.

3.4.2. *Repair Function*

As discussed in the previous sub-section, four cases cause an infeasibility of a solution. Repair function is provided for each of them.

Case #1: In this case, the repairing heuristic of ToCMA firstly discovers the origin of infeasibility, e.g., node n_i. Secondly it finds the node's nearest neighbour. Thirdly, it measures the distance to that neighbour and calculates the necessary power needed by n_i to be able to communicate with that neighbour. And finally the repairing heuristic assigns this power value to n_i.

Case #2: In this case,the heuristic firstly discovers the one way isolated groups, e.g., g_1, g_2, g_3. Then it tries to find a node n_{i_1} in group g_1 which is the nearest neighbour of a node n_j in group g_2 and also a node n_k in group g_3 which is the nearest neighbour of a node n_{i_2} in group g_1. Finally, it calculates the power needed for n_{i_1} to communicate with n_j and assigns this power value to n_{i_1} and the power needed for n_k to communicate with n_{i_2} and assigns this power value to n_k. In the special case of Case #2 the heuristic discovers the origin of infeasibility, e.g., node n_i, and it's nearest neighbour. Then, it measures the distance to that neighbour and calculates the necessary power needed by n_i to be able to communicate with that neighbour. Finally the repairing heuristic assigns this power value to n_i.

Case #3: In this case, the repairing heuristic firstly discovers the involved nodes. For each involved node n_i, it carries out the following steps: 1) it tries to find other next-hop neighbours and selects the one n_j which is nearest to it; 2) it calculates the power needed for n_i to reach n_j; 3) then it assigns this power value to n_i.

Case #4: In this case,the heuristic firstly discovers the partitioned groups, e.g., g_1, g_2. Then it tries to find a node n_j in group g_1 which is the nearest neighbour of a node n_i in group g_2. Finally, it calculates the power needed for n_i to communicate with n_j and assigns this power value to n_i. Then communication from g_2 to g_1 is set up. In the same way the communication from g_1 to g_2 can also be set up. Repeat this procedure to all partitioned groups until network connection is resumed.

The above methods are collectively implemented in $repair(c_i)$. The overall procedure of the repair function is described by the following pseudo code.

```
Input: a set of x chromosomes each representing
an infeasible solution,i.e.,$Pop_t^{IF} = \{c_1, c_2, ..., c_x\}$
and a set of $y$ chromosomes each representing a
feasible solution,i.e.,$Pop_t^F = \{c_1, c_2, ..., c_y\}$,
where$x + y = \mu$.
Output: a set of $\mu$ chromosomes each
representing a feasible solution,i.e.,
$Pop_t^F = \{c_1, c_2, ..., c_\mu\}$.
while $Pop_t^{IF} \neq \emptyset$ do
    select $c_i \in Pop_t^{IF}$;
    $c_i^F \leftarrow repair(c_i)$;
    $Pop_t^F \leftarrow Pop_t^F \cup \{c_i^F\}$; // add the new solution
    into the feasible solution set
    $Pop_t^{IF} \leftarrow Pop_t^{IF} - \{c_i\}$; // remove the repaired
    solution from the infeasible solution set.
```

In the worst case the time complexity of the repair algorithm is $O(Pop_t^{IF})$

3.4.3. *Improvement*

The feasible solutions are further improved by ToCMA. The purposes are mainly twofold as far as ToCMA is concerned. Firstly it is to provide remedies to the fact that the random power assignment used for population initialization might assign unnecessary high power to certain nodes as such leading to unnecessary power consumption. Secondly it is to avoid the situation where direct connections are too frequently used whereas there is an indirect route between two nodes. To this end, ToCMA is particularly interested in the following two types of improvements, as illustrated in Figure 4(a) and (b) respectively.

Improvement #1, as illustrated in Figure 4(a), occurs when a node's current power p_i is greater than the power needed to reach its farthest neighbour n_j. Denote the distance from n_i to n_j as d, then the maximum power requested for n_i is $p_i^{RQ} = d^\alpha$ according to Formula (1). If $p_i > p_i^{RQ}$ then p_i^{RQ} is used to replace p_i, namely, the power of this node is decreased to the power level that is just enough to reach its farthest neighbour(s).

Furthermore, if there is a route between a pair of indirectly connected nodes: n_i and n_j then there is a chance to take the benefit of Improvement #2. Improvement #2, as illustrated in Figure 4(b), occurs when the power needed for a node n_i to reach directly its farthest neighbour n_j is more than the power needed if n_i follows a route r to n_j. For instance, in Figure 4(b), the node's current power p_i needed to reach its farthest neighbour n_j, directly, is greater than the power p_r needed to reach n_j through an intermediate node n_k. Denote the distance from n_i to n_j as $d_{i,j}$, the distance from n_i to n_k as $d_{i,k}$ and the distance from n_k to n_j as $d_{k,j}$. Then $p_i = d_{i,j}^a$, the power needed for n_i to reach n_k is $p_{i,k} = d_{i,k}^\alpha$ and the power needed for n_k to reach n_j is $p_{k,j} = d_{k,j}^\alpha$. Thereby the total power needed by route $r(n_i, n_k, n_j)$ is $p_r = p_{i,k} + p_{k,j}$. If $p_r < p_i$ then p_i is replaced by $p_{i,k}$, namely, the power of this node is decreased to the power level that is just enough to reach the intermediate neighbour of less power consuming route. Note that by following a less energy consuming shortest path from sensor n_i to sensor n_j through an intermediate sensor n_k instead of using the direct longest path form n_i to n_j, extra delay might be introduced in the sensor network. A right balance or trade-off needs to be found considering the features of the application (e.g., whether it is more time critical or more energy critical) and the status of the network (e.g., the energy distribution). This paper focuses more on the energy-efficiency side. Its impact on delay needs to further investigated in our future work. The improvement function works as follows:

```
Input: a set of $\mu$ chromosomes each
representing a feasible solution,
i.e.,$Pop_t^F = \{c_1, c_2, ..., c_\mu\}$
Output: a set of $\mu$ chromosomes each
representing a possibly improved feasible
solution, i.e.,$Pop_t^{IMP} = \{c_1^{IMP}, c_2^{IMP}, ..., c_\mu^{IMP}\}$.
Step 1: set up loop, i.e., for each chromosome
$c_i \in Pop_t^F$, $i = 1$ to $\mu$ do
Step 2: improve each chromosome separately,
namely, carrying out the following procedure:
  for each gene $g_j$ in $c_i$, $j = 1$ to $n$ do
    Step 2.1: carry out Improvement #1 on $g_j$:$g_j^1 = IMP1(g_j)$;
    Step 2.2: carry out Improvement #2 on the output of Improvement #1: $g_j^2 = IMP2(g_j^1)$;
    Step 2.3:update the chromosome:replace $g_j$ with $g_j^2$ in chromosome $c_i$
Step 3: use the improved solution $c_i$ (also denoted as $c_i^{IMP}$ for expressing clarity) to replace the original $c_i$ in the solution set $Pop_t^F$(for expressing clarity the new solution set is denoted as $Pop_t^{IMP}$).
Step 4: return $Pop_t^{IMP}$.
```

The computational time complexity of the Improvement algorithm is $O(\mu.n)$.

The overall computational time complexity of the local search in the worst case is $O(\mu.n^2)$.

3.5. *Population Update and ToCMA Termination*

After local search, which involves in checking, repairing and improvement as discussed above, all the solutions are feasible and are kept in Pop_t^{IMP}. This Pop_t^{IMP} keeps only feasible solutions for one particular generation. ToCMA maintains another set of feasible solutions (also of length μ) that collects the best solutions across all generations. We denote this set as $BestInd$. A solution is evaluated based on its fitness value according to formula (2). The smaller a solution's fitness value is the better the solution is because in ToCMA the fitness represents the power consumed. The solutions in $BestInd$ will be used by GA operators to generate next generation populations. Using best parents across previous generations to create the next generation rather than only the current generation (i.e., Pop_t^{IMP}) increases the chance of creating best solutions in the next generation. The focus of this sub-section is how to generate and update $BestInd$ to its best solutions, which is described as follows.

```
Step 1: initialize $BestInd$: $BestInd = \emptyset$;
Step 2: first generation: if $t == 0$ then
  Step 2.1: $BestInd = Pop_0^{IMP}$;
  Step 2.2: Sort the solutions in $BestInd$ in an increasing order of their fitness values; get the last solution (or chromosome) into $c^{LAST}$.
  (Step2.3: GA operation to generate next generation)
Step 3: later generation: if $t! = 0$ then
  Using the solutions from the current generation to update $BestInd$.
  for each chromosome $c_i^{IMP} \in Pop_t^{IMP}$, $i = 1$ to $\mu$ do
    if $f(c_i^{IMP}) < f(c^{LAST})$ then use $C_i^{IMP}$ to replace last chromosome (namely the worst) in the $BestInd$.
    Sort the solutions in $BestInd$ in an increasing order of their fitness values;
Step 4: termination checking based on the number of generations
  if $t == \eta$ then terminate;
  else using GA operation to generate next generation; go to Step 3.
```

Several population replacement strategies have been proposed in the literature [25]. ToCMA uses the most common one, the elitism strategy. A solution from the current generation is used to update $BestInd$ only when it is better than the worst one in $BestInd$. And it is always the worst solution in $BestInd$ that is replaced. At the end of each generation, this strategy ensures that only μ chromosomes with best fitness value are kept in $BestInd$ and survive to get involved in the creation of next generation via genetic operator

The algorithm terminates after η generations. If the termination criterion is not met, the GA's operators manipulate on $BestInd$ to generate the new population.

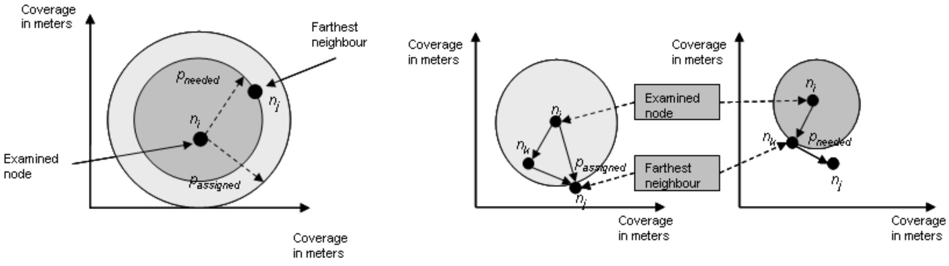
The time complexity of the algorithm is $O(log\mu)$, where μ is also the length of BestInd.

3.6. *Genetic operators:Selection,Crossover and Mutation*

Three steps are needed in order to generate a new generation while keeping its high quality and diversity: selection, crossover and mutation.

3.6.1. *Selection*

The selection operator is to improve the average quality of the population by giving the high-quality chromosomes a better chance to get copied into the next generation [22]. ToCMA uses the tournament selection [12] as its selection operator. The reason why this particular method is selected is because it only needs a preference ordering between strings as such much simpler to be applied to resource-constrained wireless networks such as sensor networks. It is efficient to code and the selection



(a) Improvement #1 *(b) Improvement #2*

Fig. 4. ToCMA Improvement Function

pressure is easily adjusted. The preference ordering in ToCMA is already carried out when generating $BestInd$.

In order to utilize the tournament selection, the tournament size m needs to be decided. In ToCMA the tournament size m is equal to $(\mu/2) + 1$(note that if μ is an odd number then the fraction part is ignored). A constant value for m, which is the case in ToCMA, renders a simpler operation of the tournament selection algorithm [12]. These m chromosomes are the first m best individuals stored in $BestInd$. The purpose of the selection operator is to get a pair chromosomes which are forwarded for further crossover operation.

The selection operator works as follows:

Input: a set with the m best integer strings
Output: a pair of integer strings.
Step 1: Set the parameter m equal to $(\mu/2) + 1$.
Step 2: Select the m first chromosomes from $BestInd$, which are also the m best chromosomes.
Step 3: While the size of new population Pop_{t+1} is less than μ, run the "tournament" and select the best two individuals based on their fitness.
  Step 3.1: Forward this pair of chromosomes to the crossover operator as parents.
  Step 3.2: Randomly select one of the selected pair and delete it from the tournament.

The computational time complexity of the algorithm is linear to μ.

3.6.2. *Crossover*

In this phase ToCMA algorithm utilizes a fast and simple technique, the single-point crossover operator, as illustrated in Figure 5.

This operator involves two steps: 1) the selection of the crossover site $1X$, and 2) the generation of the two new chromosomes (also called offsprings).

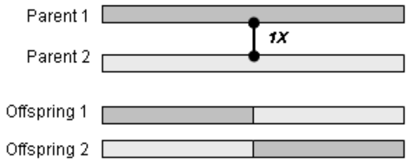


Fig. 5. Single Point Crossover

The crossover site is selected randomly in the interval $[1, n]$. Offsprings are generated by swapping the characters between positions $1X + 1$ and n of the parents (the pair selected earlier by the selection operator). For example, the following two parent chromosomes are selected - each of length $n = 7$:

$$c1: \ 1\,3\,4\,6 \,|\, 9\,2\,5$$
$$c2: \ 8\,1\,7\,5 \,|\, 3\,2\,8$$

Let the crossover site be 4. The two substrings between 5 and 7 are swapped and the two substrings between 1 and 4 remain the same. The two offsprings generated are as follows:

$$o1: \ 1\,3\,4\,6 \,|\, 3\,2\,8$$
$$o2: \ 8\,1\,7\,5 \,|\, 9\,2\,5$$

The crossover will continue until a new population Pop_{t+1}^{CRS} is generated with μ offspring. This is also a linear algorithm. The crossover operator works as follows:

Input: a pair of integer strings as parents
Output: a pair of integer strings as offspring.
Step 1: Accept the pair of chromosomes from the selection operator and set them as the parents.
Step 2: Randomly generate a number between the interval $[1, n]$ and set it as the crossing site $1X$.
Step 3: Swap the characters between positions $1X + 1$ to n of the parents to generate the two offspring.
Step 4: Add the two offspring in the set of the new population Pop_{t+1}^{CRS}.

The problem with crossover is that all the populations generated tend to gather together as such covering only a limited area in the search space. In order to increase the diversity of the population and thereby increasing the chance of finding better solutions, mutation is further utilized after crossover.

3.6.3. *Mutation*

The mutation operator is applied to all offsprings of the new population Pop_{t+1}^{CRS}. This operator simply selects randomly g genes in each offspring and randomly changes their value in the power interval $[0, \Delta]$. The number of genes that will be mutated is $g = \lfloor n/2 \rfloor$ (round the fraction up if n is an odd integer), i.e., half of the nodes. This number is chosen to be big enough to increase the diversity of the population as much as possible in order to compensate the diversity loss in crossover operator and the local search process [14].

For example, an original offspring of length $n = 7$:

$$Original\ offspring:\ 1\ 3\ 4\ 6\ 3\ 2\ 8$$

The number of genes to be mutated is $g = \lfloor n/2 \rfloor = 4$. Let the positions of genes that are randomly selected for mutation be $(1, 3, 4$ and $7)$. Then after mutation we have:

$$Mutated\, Offspring:\ \mathbf{5}\ 3\ \mathbf{7}\ \mathbf{8}\ 3\ 2\ \mathbf{1}$$

After the mutation of all μ individuals of the new population Pop_{t+1}^{CRS}, a new mutated population $Pop_{t+1}^{MUT}$is created. This new population Pop_{t+1}^{MUT} will then be forwarded to the local search heuristic discussed above to go through checking/repairing/improvement again until the termination criterion is met.

The mutation operator, which is a linear process, works as follows:

```
Input: a set of $\mu$ chromosomes. $Pop_{t+1}^{CRS}$
Output: a set of $\mu$ chromosomes.$Pop_{t+1}^{MUT}$
Step 1: Pick each offspring from the new population $Pop_{t+1}^{CRS}$ one by one.
Step 2: For each offspring $o_i \in Pop_{t+1}^{CRS}$
    Step 2.1: Calculate $g = \lfloor n/2 \rfloor$ and randomly choose $g$ genes of the current chromosome to change their values.
    Step 2.2: Replace the current values of the selected $g$ genes with a randomly selected number within the interval $[0, \Delta]$.
Step 3: Add the mutated offspring to the set $Pop_{t+1}^{MUT}$.
```

4. Evaluation Results and Analysis

In this section ToCMA is compared against MST (minimum spanning tree) in terms of total energy consumption while maintaining network connectivity. MST is selected because it is a simple and neat solution to the MENC problem and it is also popularly selected as a benchmark in topology control problem solving [3–5,26].

4.1. *Experimental Design*

A spanning tree, $T(V, E')$, is a subgraph induced from a supergraph $G(V, E)$, where V is a set of nodes common for both graphs, E is a set of links in the supergraph and $E' \in E$ is a set of links in the subgraph. The number of links in the subgraph is equal to $|E'| = |V| - 1$. A graph may have more than one spanning trees. From all these spanning tree, there is one that has minimum cost, based on the sum of the weight on links. This is called the minimum spanning tree T of graph G [10,11]. The same method as that in [7] is utilized to calculate the energy consumption of MST.

The number of chromosomes created at each generation is $\mu = 30$ and the number of nodes n used in different networks varies from 10 to 100. ToCMA employs a tournament selection with tournament size m= 16. A single point crossover is then performed to each pair of chromosomes. The crossing site is randomly chosen from the interval $[1, n]$. A mutation is carried out to all offspring with mutation rate being $\lfloor n/2 \rfloor$. Each node is randomly assigned a power within the interval $[0, 500^{\alpha}]$, where the path loss exponent $\alpha = 2$. The best $\mu = 30$ solutions from all the generations are stored in an increasing order in $BestInd$. Each experiment is terminated when the number of generations reaches the maximum number of $\eta = 15$.

A number of stationary nodes were randomly deployed on a two-dimensional plane (500x500), so the power that could be randomly assigned in ToCMA at the beginning is between the boundaries of 0-500^{α}. The size of the network (number of nodes) was the same for both algorithms and the fitness function (total energy consumption of the network) is then measured.

For small networks, a search is performed for both MST and ToCMA, the results of which are shown in Table 2. This table shows a clear performance win of ToCMA over MST. And for cases when n=3-6

ToCMA can approach the optimal solutions.

Table 2
Comparison of MST and ToCMA for Small Networks

Number of nodes	ToCMA	MST
3	915	960.
4	1000	1070.
5	1550	1605.
6	1020	1200.
7	2555	2978.5.
8	2848	2988.
9	3208.5	3305.
10	3710	3970.

For big networks of large number of nodes, simulation is used. We run MST and ToCMA algorithms separately for networks of different density and draw the results in Figure 6. Figure 6 gives the total energy consumption of the best solutions found by each algorithm. The energy consumption used by y-axis is represented by a ratio of e_B to ω . e_B stands for the total energy consumption of the best solution and is calculated via fitness function (Formula 2).ω stands for the total energy consumed when each node uses its maximum power (Δ), i.e., $\omega = n \times \Delta^\alpha$ where $\alpha = 2$. If we denote the ratio as ra then we have

$$ra = e_B/\omega = \sum_{i \in BestSol} p_i/(n \times \Delta^\alpha) \qquad (3)$$

It is observed that MST and ToCMA keep a similar curve shape as the number of nodes in the network increases. Furthermore, the total energy does not change too much as the node number increases from 10 to 100. This is because increased node density reduces the communication distance between neighbouring nodes and as a consequence reducing the transmission power of each node. Note that in most cases the ToCMA algorithm outperforms MST algorithm.

This result is also depicted by the following example as illustrated in Fig. 8. In this example a random topology is created and the links between the nodes are calculated based on the location of the nodes and their power assignment.

Figure 7 exemplifies how ToCMA outperforms MST using a specific network topology. In Figure 7 (a) MST creates a strongly connected network by using in an increasing order the shortest links of the network. This is indicated by the bold lines. The dotted lines indicate the links that are not used. The total energy consumption by MST is:
$e_B(MST) = \sum_{i=1}^{8} p_i = 4^2 + 4^2 + 4^2 + 5^2 + 1^2 + 5^2 + 5^2 + 5^2 = 149$
Similarly, for ToCMA, there is:
$e_B(ToCMA) = \sum_{i=1}^{8} p_i = 1^2 + 1^2 + 1^2 + 5^2 + 5^2 + 5^2 + 5^2 + 5^2 = 128$.
Apparently, $e_B(ToCMA) < e_B(MST)$.

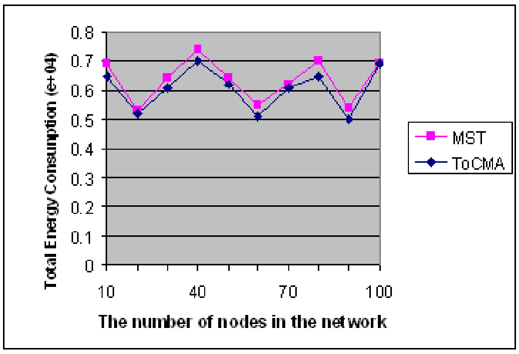
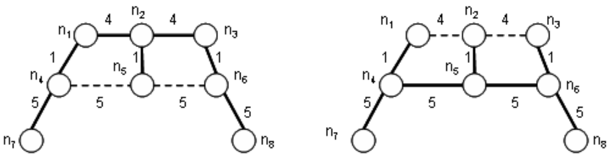


Fig. 6. Comparison of MST and ToCMA for Big Networks



(a) - MST power assignment

(b) - ToCMA power assignment

Fig. 7. An Example

4.2. *Other Benefits & Discussions*

Figure 6 indicates a narrow win of ToCMA over MST. In addition to the better quality of solution that ToCMA can offer, ToCMA can also provide some other benefits.

Table 3
The *BestInd* set (for α=2)

BestInd									
Chromosome	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	**Total**
Best - c_1	1^α	1^α	1^α	5^α	5^α	5^α	5^α	5^α	128
c_2	4^α	4^α	1^α	5^α	5^α	5^α	5^α	5^α	142
c_3	1^α	4^α	1^α	5^α	5^α	5^α	5^α	5^α	143
c_4	4^α	4^α	4^α	5^α	1^α	5^α	5^α	5^α	149
...	...	...	...	...	...	...	...	...	...

ToCMA can take advantage of the information stored in *BestInd* that is generated during update function (refer to sub-section 3.5) for a variety of purposes. In this set the best solutions are stored in an increasing order of total power consumption, as illustrated in Table 3. Apart from being effectively used for genetic operations, $BestInd$, when stored in the sink of the sensor network, can be utilized, for example, for fault tolerance purpose. Table 3 illustrates how it works. For example, if for any reason a sensor node has to operate with less power than the one assigned by the best solution c_1, then the sink can search in $BestInd$ to try to find another best solution that satisfies this new power constraint requirement. And then the sink broadcasts this solution to all the sensors in the network to easily accommodate a sudden change in the network. For instance, after solution c_1 has been deployed, suddenly node n_4 has to operate with a power $p_4 \leq 3^\alpha$, then ToCMA can easily locate the best solution for this situation, i.e., c_4. This can potentially reduce the chance of network partitioning.

In order to check that a solution provides a strongly connected network, the checking function in Section 3 actually needs to guarantee a route, either directly connected or via intermediate nodes, between any pair of nodes in the network. This routing information can be well stored alongside the power information (for instance, in an extended $BestInd$ table), and as such evolves cross generations. With this routing information, route discovery process can be much simplified. An energy-aware routing algorithm supported by ToCMA and its performance analysis against other mainstream energy-aware routing algorithms are our near-future work. Though being more computationally complex than MST, the outputs from ToCMA can be utilized for many other purposes than energy consumption efficiency to the overall benefit of the wireless network as a whole.

5. Conclusions and Future Works

In this paper, we have proposed an alternative approach to tackle the MENC problem in wireless sensor networks, which utilizes modern heuristics and more precisely memetic algorithm. The proposed ToCMA explores in an effective manner the solution space by using a combination of the genetic algorithm operators and the local search technique. It employs repair and improvement methods to refine solutions. The concrete MA solutions are guided by problem-specific features such as network connectivity, avoiding loop etc. Simulation results have shown that better solutions can be obtained by ToCMA than MST. ToCMA also demonstrated its strength in generating initial routing information and fault tolerance and robustness.

Based on the encouraging results of this paper, we will further investigate how an intelligent initialization and smarter local search mechanism will make impact on quality solutions and the overall performance of the algorithm. To encode routing information into chromosome on top of power value and to design energy-constrained routing algorithm using MA is also within our future work. Currently the choice of mutation position in a chromosome is made randomly. A guided mutation based on the feature of the network topology and energy assignment is to be investigated. To further reduce the computational complexity is also the next-step target when accommodating the above future research plans.

Acknowledgement

The work of this paper was partially funded by the UK EPSRC (Engineering and Physical Sciences Research Council) under the Project PANDA (EP/D061881/1), and partially funded by the RPF (Research Promotion Fund) 2006 of University of Essex.

References

[1] I. Akyildiz, W. Su, and Y. Sankarasubramaniam, A Survey on Sensor Networks, IEEE Communication Magazine, August 2002, pp. 102-114.

[2] A. Warrier, S. Park, J. Min and I. Rhee, How much energy saving does topology control offer for wireless sensor networks? - a practical study,Department of Computer Science North Carolina State University Raleigh, NC, 2006

[3] I. Stojmenovic, M. Seddigh, J. Zunic, Dominating Set and Neighbour Elimination-Based Broadcasting Algorithms in Wireless Networks, IEEE transactions on parallel and distributed systems, Vol 13, No.1, Jan.2002.

[4] I. Stojmenovic, Position based routing in ad hoc networks, IEEE Commmunications Magazine, Vol. 40, No. 7, July 2002, 128-134.

[5] C. Schurgers and M. B. Srivastava, Energy Efficient routing in wireless sensor networks, Networked & Embedded Systems Lab (NESL), Electrical Engineering Department University of California at Los Angeles (UCLA), CA, 2001.

[6] D. Ganesan, R. Govindan,S. Shenker and D. Estrin, Highly-Resilient, Energy-Efficient Multipath Routing in Wireless Sensor Networks, Mobile Computing and Communications Review, Vol. 1, No. 2.

[7] M. X. Cheng et al., Topology Control of Ad Hoc Wireless Networks for Energy Efficiency, IEEE Transactions on Computers, vol. 53, no. 12, December, 2004.

[8] X. Cheng et al, Strong Minimum Energy Topology in Wireless Sensor Networks: NP-Completeness and Heuristics, IEEE Transactions on Mobile Computing, vol. 2, no. 3, 2003.

[9] A. K. Das, et al., r-shrink: A Heuristic for Improving Minimum Power Broadcast Trees in Wireless Networks.

[10] W. J. Cook et al, Optimal Trees and Paths: Minimum Spanning trees, Combinatorial Opt., ch. 2, pp. 9-19, 1998.

[11] J. Wieselthier, G.D. Nguyen, and A. Ephremides. On the construction of energy-efficient broadcast and multicast trees in wireless networks. In Proceedings of the IEEE Infocom 2000, pages 585 - 594, Tel Aviv, Israel, 2000.

[12] C. Reeves, Genetic Algorithms. In Handbook of Metaheuristics, ch. 3, pp. 65-67.

[13] D. R. Din and Shian-Shyong, Genetic Algorithms for Optimal Design of the Two-Level Wireless ATM network, Proc. Natl. Sci. Counc. ROC(A), Vol. 25, No. 3, pp. 151-162, 2001.

[14] P. Moscato, A Gentle introduction to Memetic Algorithms. In Handbook of Metaheuristics, ch. 5, pp. 105-143.

[15] Q. Zhang and Yiu-Wing, An Orthogonal Genetic Algorithm for Multimedia Multicast Routing, IEEE Transactions on Evolutionary Computation, Vol. 3, No. 1, April 1999.

[16] H. E. William, E. N. Krasnogor and J. E. Smith (eds), Recent Advances in Memetic Algorithms, Springer, 2005.

[17] D. Corne, M. Dorigo and F. Glover (eds), New Ideas in Optimization, McGraw-Hill Education, 1999.

[18] H. Ishibuchi and T. Murata, A Multi-Objective Genetic Local Search Algorithm and Its Application to Flowshop Scheduling, IEEE Trans. On Systems, Man and Cybernetics - Part C, Vol. 28, No. 2, pp. 99-110, 2004.

[19] P. Merz and B. Freisleben, Fitness landscape analysis and memetic algorithms for the Quadratic Assignment Problem, IEEE Trans. On Evolutionary Computation, Vol. 3, No. 1, pp. 53-62, 1999.

[20] Y. Xiong, B. Golden and E. Wasil, A One-Parameter Genetic Algorithm for the Minimum Labelling Spanning Tree Problem, IEEE Transactions on Evolutionary Computation, Vol. 9, No. 1, February 2005.

[21] J. J. Car, Directional or omnidirectional antenna?, Technote no. 1, Universal Radio research.

[22] C. W. Ahn and R. S. Ramakrishna, A Genetic Algorithm for Shortest Path Routing Problem and the Sizing of Populations, IEEE Transactions on Evolutionary Computation, Vol. 6, No. 6, December 2002.

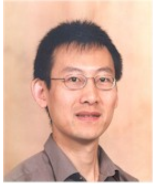
[23] J. Rees and G.J. Koehler, An investigation of GA performance results for different cardinality alphabets. In Evolutionary Algorithms: IMA Volumes in Mathematics and its Applications, vol. 111, Springer-Verlang, 191-206, 1999.

[24] X.Hue, Genetic algorithms for optimization:Background and applications. Edinburg Parallel Computing Centre, Univ. of Edinburg, Edinburg, Scotland, Ver. 1.0, Feb. 1997.

[25] D. Lawrence, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, U.S.A, 1991.

[26] M. Cardei, J. Wu and S. Yang, Topology Control in Ad Hoc Wireless Networks using Cooperative Communication,Proc. of IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON), 2005.

Andreas Konstantinidis received the Higher Diploma of Technician Engineer in the Electrical Engineering department of Higher Technical Institute, Nicosia, Cyprus, the BSc with honors in Electrical Engineering Computer Engineering of Electron Devices department, Budapest University of Technology and Economics (BUTE) and the MSc with distinction in Computer and Information Networks of Electronic Systems Engineering department, University of Essex. He is currently a PhD student in Computer Science and Electronic Systems Engineering departments at the University of Essex. His current research interests are wireless networking with focus on wireless ad hoc and sensor networks, evolutionary computation, combinatorial optimization and heuristic search.
Email: akonst@essex.ac.uk



Kun Yang has been working as an academic staff in the Department of Electronic Systems Engineering, University of Essex, UK since 2003. Before that he worked for 3 years at the Department of Electronic and Electrical Engineering, University

College London (UCL) - UK on several European Union IST projects, such as MANTRIP, FAIN, CONTEXT in a wide area of networking, network and service management. He received his PhD from the Computer Science Department of Jilin University, China in the area of distributed artificial intelligence. His MSc and BSc degrees were both from Jilin University, China in the field of distributed systems and computer networks in 1994 and 1991 respectively. His main research interests include telecommunications and computer networks, wireless networks, mobile networks and mobile computing, mobile ad-hoc and sensor networks, multimedia over wireless networks, and pervasive service engineering. He has published journal papers and co-authored book chapters, amongst other conference papers. He serves on the editorial board of IEEE Comm. Surveys and Tutorials, John Wiley Journal of Wireless Communications and Mobile Computing (WCMC), and serves as a special issue guest editor in international journals. He is a chair (e.g., ICPS06) or a TPC member of many international conferences (e.g., ICC07 and Globecom06). He is a member of IEEE, IEE and ACM.
Email: kunyang@essex.ac.uk

Hsiao-Hwa Chen is currently a full-Professor, National Sun Yat-Sen University, Taiwan. He has been a visiting Professor to Department of Electrical Engineering, University of Kaiserslautern, Germany, in 1999, the Institute of Applied Physics, Tsukuba University, Japan, in 2000, Institute of Experimental Mathematics, University of Essen, Germany in 2002 (under DFG Fellowship), Department of Information Engineering, The Chinese University of Hong Kong, 2003, and Department of Electronics Engineering, The City University of Hong Kong, 2006. His current research interests include wireless networking, MIMO systems, and next generation CDMA technologies for future wireless communications. He has authored or co-authored over 160 technical papers in major international journals and conferences, and five books in the areas of communications, including "Next Generation Wireless Systems and Networks" (498 pages) published by John Wiley and Sons. He served or is serving as International Steering Committee member, Symposium Chair and General Co-Chair of more than 50 major international conferences, including IEEE VTC, IEEE ICC, IEEE Globecom, and, IEEE WCNC, etc. He served or is serving as the Editor, Editorial Board member or Guest Editor of many international journals, including IEEE Communications Magazine, IEEE JSAC, IEEE Vehicular Technology Magazine, IEEE Wireless Communications Magazine, IEEE Transactions on Wireless Communications, Wiley's Wireless Communications and Mobile Computing (WCMC) Journal, Wiley's International Journal of Communication Systems, etc. He has been a Guest Professor of Zhejiang University, China, and Shanghai Jiao Tung University, China, since 2003 and 2005 respectively.
Email: hshwchen@ieee.org

Qingfu Zhang received the BSc in mathematics from Shanxi University, Shanxi, China in 1984, the MSc in applied mathematics and the PhD in information engineering from Xidian University, Xi'an, China, in 1991 and 1994, respectively. He is now a Reader in the Department of Computer Science, University of Essex, UK. From 1994 to 2000, he worked in the National Laboratory of Parallel Processing and Computing in National University of Defence Science and Technology (China), Hong Kong Polytechnic University (Hong Kong), the German National Research Centre for Information Technology (now Fraunhofer-Gesellschaft, Germany) and University of Manchester Institute of Science and Technology (UK). His main research areas are evolutionary computation, optimisation, neural networks, data analysis and telecommuication networks. He is an associate editor of IEEE Transactions on Systems, Man, and Cybernetics-Part B.
Email: qzhang@essex.ac.uk