Selecting the Best Spanning Tree to Reduce the Interference of a wireless sensor network using Genetic Algorithm

Vahid Haghhighatdoost  
Department of Computer Engineering,  
Shahed University, Tehran, Iran

Maryam Espandar  
Department of Computer Engineering,  
Shahed University, Tehran, Iran

Abstract
The interference reduction is one of the most important problems in the field of wireless sensor networks. Wireless sensor network elements are small mobile receiver and transmitters. The energy of processor and other components of each device is supplied by a small battery with restricted energy. One of the meanings that play an important role in energy consumption is the interference of signals. The interference of messages through a wireless network, results in message failing and transmitter should resend its message, thus the interference directly affects the energy consumption of transmitters. This paper presents an algorithm which suggests the best spanning tree for the input distribution of the nodes in the plane how the interference of the network aims the minimum value.

Keywords: Genetic algorithm, interference, sensor network, spanning tree, wireless ad-hoc network.

1. Introduction
Wireless sensor networks consist of mobile nodes equipped with, among other components, a processor, some memory, a wireless radio, and a power source. Due to physical constraints, nodes are primarily powered by a weak battery, so energy is a scarce resource in wireless ad-hoc networks.

In a general way, topology control can be considered as the task of, given a network communication graph, constructing a spanning tree while minimizing energy consumption. Additionally, symmetric links are desired as they permit simpler higher-layer protocols [1]. One of the foremost approaches to achieve substantial energy conservation is by minimizing interference between network nodes.

The concept of topology control restricts interference by reducing the transmission power levels at the network nodes and cutting off long-range connections in a coordinated way. At the same time transmission power reduction has to proceed in such a way that the resulting topology preserves connectivity.

The intuition was that a low minimizing the maximum degree of nodes of graph would solve the interference issue automatically, and as depicted in [1] this intuition was proved wrong in [2]. The general interference model introduced in [3], proposes a natural way to define interference in ad-hoc networks. The general question is:

How can one connect the nodes such that as few nodes as possible disturb each other? In the following, we discuss the network and interference model presented in [3].

A geometric graph is used for modelling of the wireless network. The graph consists of a set of nodes represented by points in the Euclidean plane; we want to connect these nodes by choosing a set of symmetric edges. A node is able to adjust its transmission power to any value between zero and its maximum power level to reach other nodes. An edge exists if and only if the maximum transmission range of both incident nodes mutually includes their counterpart. The minimum requirement of a topology control algorithm is reducing transmission power to compute a subgraph of the given network graph that preserves connectivity. The interference of a node \( v \) is then defined as the number of other nodes that potentially affect message reception at node \( v \). The maximum interference of a graph is then defined as the maximum node interference.

So far, not many results have been published in the context of explicit interference minimization. For networks restricted to one dimension the authors in [3] present a \( \sqrt{n} \)-approximation of the optimal connectivity preserving topology that minimizes the maximum interference. For the two dimensional case, the authors in [4] propose an algorithm that bounds the maximum interference to \( O(\sqrt{n}) \).

A theoretical problem in topology control which has been stated as essential to understanding sensor networks is the following:

Given \( n \) nodes in the plane, connect the nodes by a spanning tree. For each node \( v \) we construct a disk that its center is located on node \( v \) with radius equal to the distance to \( v \)'s furthest neighbour in the spanning tree.

The interference of a node \( v \) is then defined as the number of disks that include node \( v \). Find a spanning tree that minimizes the maximum interference.

Kevin Buchin in [5] proved that the interference reduction problem is NP-complete. If we have \( n \) nodes and we want to find the best spanning tree we should generate \( N^{N^2} \) different trees; it means for \( n=12 \) we should generate 61,917,364,224 trees and for a network with \( n=20 \), number of trees become 2.62144E+023. In a similar work,
Faghani and his colleagues in [6] used the genetic approach to find the best spanning Tree in metro Ethernet networks. In this paper Faghani's solution is extended and applied to solve the interference problem.

2. Interference model of network

The network is modelled as a geometric graph $G=(V,E)$. Let $N_u$ denote the set of all neighbours of a node $u \in V$ and $r_u$ determines the distance from $u$ to its farthest neighbour. More precisely $r_u = \max_{v \in N_u} |u-v|$, where $|u-v|$ denotes the Euclidean distance between nodes $u$ and $v$. $D(u, r_u)$ denotes the disk centered at $u$ with radius $r_u$ covering all nodes that are possibly affected by message transmission of $u$ to one of its neighbours. Then the interference of a node $v$ is defined as the number of other nodes that potentially affect message reception at node $v$.

**Definition 1:** Given a graph $G=(V,E)$, the interference of a node $v \in V$ is defined as:

$$I(v) = \#\{u \in V \setminus \{v\}, v \in D(u, r_u)\}$$

(1)

Note that even though each node is also covered by its own disk, we do not consider this kind of self-interference. The graph interference is the maximum interference occurring in a graph:

**Definition 2:** The interference of a graph $G=(V,E)$ is defined as:

$$I(G) = \max_{v \in V} I(v)$$

(2)

Figure 1. The interference model of a graph with 5 vertexes

As shown in Figure 1 the interference of nodes is as follow:

<table>
<thead>
<tr>
<th>Node</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interference</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

According to Definition 2 the Interference of graph $I(G)=2$.

3. The nearest neighbour forest

In the first view of the interference problem, one may say the nearest neighbour forest or minimum spanning tree is the best subgraph which results in minimum interference. In this section, it is shown that this is already a substantial mistake, as thus interference becomes asymptotically incomparable with the interference-minimal topology.

Figure 2. Two Special node distributions and resulting topology by applying the AMST algorithm
For some special distribution, the nearest neighbour forest results in the worst interference. Clark et al., in [7], introduced an instance which seems to yield inherently high interference: the so-called exponential node chain is a one-dimensional graph $G=(V,E)$ where the distance between two consecutive nodes grows exponentially from left to right as depicted in Figure 2(a). That is, the distance between nodes $v_i$ and $v_{i+1}$ is $2^i$ for $i=0,1,2,\ldots,n-1$. So as shown in Figure 2(c), the nearest neighbour forest results in the interference of $\Omega(n)$. Also authors in [3] introduced the Two Exponential node chains as shown in Figure 2(b), on the bottom, there is a horizontal chain of nodes $v_i$ with exponentially growing distances, the same as the one-dimensional exponential chain, thus distance between $v_i$ and $v_{i+1}$ is $2^i$. Each of these nodes $v_i$ has a corresponding node $t_i$ vertically displaced by a little more than $v_i$’s distance to its left neighbour, that is, $|v_i - t_i| > d_i$, where $d_i = |v_i - v_{i+1}| = 2^{-i+1}$. Note that the nodes $t_i$ also form a (diagonal) exponential node chain. Finally, between two of these diagonal nodes $t_{i-1}$ and $t_i$, an additional helper node $c_i$ is placed such that $|v_i - c_i| \geq |v_i - t_i|$. The Nearest Neighbour Forest for this node distribution is shown in Figure 2(b).

Figure 3 but there is no algorithmic method which generates automatically similar subgraph.

Figure 4 shows the $A_{\text{exp}}$ algorithm resulting topology for exponential chain which is proposed in [3].

4. Genetic algorithm approach

Genetic Algorithm (GA) is a search technique used to find the approximate solution. Moreover, it improves all potential solutions step by step through biological evolutionary processes like crossover, mutation, etc. Because the process in the GA approach is not wholly operated randomly but includes both directed and stochastic search embedded with a survival of the fittest mechanism, it is possible to enforce the search to reach the optimal solution.

In [8], a genetic algorithm is proposed for degree-constrained Minimum Spanning Tree problem; and in [6], the authors used the idea proposed in [8] to develop a new genetic algorithm for selecting the best spanning tree in Metro Ethernet networks based on load balance criterion.

In our paper, we use a similar idea proposed in [6] to develop a new genetic algorithm for selecting the best spanning tree to connect the nodes in a wireless sensor network distributed in the plane to aim the minimum interference through the network.

For each GA based solution, it is necessary to define:

- The individual on which it operates (encoding),
- The operators it uses,
- Some parameters such as the population size, etc.,
- An objective function.
A. Chromosome Representation (encoding)

For GA approach, it is important to determine the adequate chromosome representation of problem. One of the classical theorems in graphical enumeration is Cayley’s theorem [9]. It states that there are $N^{N-2}$ distinct labeled trees for a complete graph with $N$ vertices. Prüfer provided a constructive proof of Cayley’s theorem establishing a one-to-one correspondence between such trees and the set of all permutation of $N-2$ digits. This means we can describe our tree with $N-2$ uniquely digits for $N$ vertices. The sequence of digits is named the Prüfer Number in [6]. Figure 5 shows two different simple trees and their Prüfer numbers. In Our GA approach the Prüfer number is used as a chromosome.

![Figure 5: Two trees with their corresponding Prüfer Number](image)

The Prüfer number encoding procedure is as below:

**Step 1)** Let $i$ be the smallest leaf node and node $j$ be incident to node $i$. Set $j$ be the first digit in the encoding. The encoding is built by appending digits to the right.

**Step 2)** Remove node $i$ and the edge from $i$ to $j$.

**Step 3)** Repeat above operation until only one edge is left.

In a Prüfer number encoding, a tree is encoding as a Prüfer vector $P$ and a set of its eligible nodes $P$ (the set of all node not included in $P$).

The decoding procedure is as below:

**Step 1)** Let node $i$ be the smallest eligible node of $P$ and node $j$ be the leftmost element of $P$. If $i \neq j$, add the edge $(i,j)$ into the tree $T$. If $i$ is no longer eligible, then remove node $i$ from $P$. Delete $j$ from $P$. If $j$ does not occur anywhere in the remaining part of $P$, then put it into $P$. Repeat the process until $P$ is empty.

**Step 2)** For the remaining last two nodes $u$ and $v$ of $P$, add the edge $(u,v)$ into the tree $T$.

Genotypes (chromosome values) are uniquely mapped on to the decision variables (phenotypic) domain.

B. Crossover and Mutation

The genetic algorithm uses the individuals in the current generation to create the children that make up the next generation. Besides elite offspring, the individuals in the current generation with the best fitness values, the algorithm creates:

- Crossover offspring by selecting vector entries, or genes, from a pair of individuals in the current generation and combines them to form a child.
- Mutation offspring by applying random changes to a single individual in the current generation to create a child.

Crossover and Mutation are two deterministic operators in the biological evolutionary process. There are several types of crossover operator such as: single point, two point, uniform, and etc. In this paper, we use single point crossover operator. Single point crossover at first generates a random position $R$ and then selects genes 1 to $R$ from parent 1 and genes $R+1$ to $N$ from Parent 2 where $N$ is the length of each chromosome. Figure 6 shows the crossover operation.

![Figure 6: Crossover operation, topology of offspring are displayed under of their Prüfer Numbers](image)

Also Mutation operator applies random changes in some chromosomes to avoid that we will not be placed in local minimum.

![Figure 7: Mutation operation, topology of offspring is displayed under of its Prüfer Number](image)
mutation operator. Figure 7 illustrates the mutation operation where a random position is selected first and its digit is replaced with another random digit.

C. Initial Population, Evaluation and Selection Operator

As the Minimum spanning tree (MST) of a graph presents the connectivity with smallest weight of graph, so we add the MST tree to the initial population. To evaluate our population, in GA approach, we need an evaluation function. We name it $Eval(X)$, in which $X$ is the Chromosome (In our research is Prüfer number) and it is called fitness value for chromosome $X$. The main goal is to minimize the $Eval(X)$. The evaluation function is used to select the best chromosomes from the population why that chromosomes with higher fitness value will have more chance to be selected for next generation. In this paper, we use Roulette wheel as the selection operator. Each slice in Roulette wheel is proportional to its fitness value.

As expected we use the graph interference for evaluation function.

$$
D(u,r,v) = \sum_{i=1}^{n} \min(d(v_i),r) \quad \text{for } v_i \in D(u,r) \cap V
$$

Where the function Corresponding_Tree returns the adjacent matrix of a graph according to its Prüfer number and $D(u,r,v)$ determines the set of nodes that are located in the disk graph centered by $u$ with radius $r$; and $r$ is the distance of farthest adjacent of node $u$.

D. Proposed GA method

The proposed GA method is outlined as follows:

1. **Step 1) Initialization:** Choose the population size $N$, proper crossover probability $P_c$ and mutation probability $P_m$, and Generate initial population $P(0)$. Let the generation number $t=0$.

2. **Step 2) Crossover:** Choose the parents from $P(t)$ with probability $P_c$ for crossover. Afterwards, randomly match every two parents as a pair and use the proposed crossover operator to each pair to generate two offspring. All offspring constitute a set denoted by $S_c$.

3. **Step 3) Mutation:** Selection the parents for mutation from set $S_c$ with probability $P_m$. For each chosen parent, the proposed mutation operator is applied to it to generate a new offspring. These new offspring are replaced with their parents in $S_c$ and constitute a set denoted by $S_m$.

4. **Step 4) Selection:** Select the best $N$ individuals among the set $P(t) \cup S_m$ as the next generation population $P(t+1)$ using Roulette wheel method, let $t=t+1$;

5. **Step 5) Termination:** If termination conditions hold, then stop, and keep the best solution obtained as the approximate global optimal solution of the problem; otherwise, go to step 2. Selected values for the above GA are as follow:

<table>
<thead>
<tr>
<th>N</th>
<th>$P_c$</th>
<th>$P_m$</th>
<th>Max Generations</th>
<th>Stall Gen Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>60</td>
<td>0.5</td>
<td>0.2</td>
<td>500</td>
<td>150</td>
</tr>
</tbody>
</table>

5. Simulation results

The most complex part of proposed algorithm is decoding the chromosomes and calculating of their fitness; so, the computational complexity of our algorithm is:

$$O(TN^2M^2) = O(TNM^4)$$

Where $T$ is the max iteration count and $N$ is population size and $M$ is number of network nodes. All simulations were done in MATLAB R2009a on a computer with 2GB of RAM and an Intel(R) Core(TM)2 T 5870 CPU. Figure 8 shows a random distribution of 10 and 20 nodes and suggested topologies with $A_{MST}$ and $A_g$.
displays the resulting topology by using the $A_{MST}$ and $A_g$ for random distribution of nodes in the plane. Figure 9 illustrate the resulting topology by using the MST and $A_g$ and $A_{exp}$ for exponential node chain distribution. Table 2 shows the final interference of different algorithms with time complexity of Genetic approach.

![Suggested topologies for different algorithms](image1)

**Figure 9:** Exponential node chain for 10 and 20 nodes. Some edges are depicted as arcs and x dimension is shown in logarithmic scale.

It is true that the result of $A_g$ for Exponential node chain and Two exponential chain is not the best topology but notice that the proposed $A_g$ is an applicable solution; also it’s exressible that the result topology of $A_g$ is acceptable for both distributions.

6. Conclusion

As proved in [5] the Interference Minimization is an NP-Complete problem and already it is introduced as an open problem in [1]. So finding the best topology with minimum interference is impossible when the nodes are more than 20. In this paper, we introduced a new Genetic Algorithm approach for finding the best spanning tree for the input distribution of wireless sensor network. We select the best tree based on the interference of resulting topology. We used the Prüfer number for encoding the individuals. It seems that the proposed algorithm could be used for distribution of nodes in the space (means in three dimensions).
Table 2- Different distributions and the interference of each one using three different algorithms MST, A\text{exp} and A\text{ag}.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Nodes count</th>
<th>MST interference</th>
<th>A\text{exp} Interference</th>
<th>A\text{ag} Interference</th>
<th>A\text{g} Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D Random</td>
<td>10</td>
<td>4</td>
<td>-</td>
<td>3</td>
<td>14.2344</td>
</tr>
<tr>
<td>2D Random</td>
<td>20</td>
<td>5</td>
<td>-</td>
<td>4</td>
<td>27.3906</td>
</tr>
<tr>
<td>2D Random</td>
<td>50</td>
<td>6</td>
<td>-</td>
<td>5</td>
<td>78.1406</td>
</tr>
<tr>
<td>Exponential chain</td>
<td>10</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>11.3594</td>
</tr>
<tr>
<td>Exponential chain</td>
<td>20</td>
<td>18</td>
<td>6</td>
<td>8</td>
<td>19.9219</td>
</tr>
<tr>
<td>Exponential chain</td>
<td>50</td>
<td>48</td>
<td>10</td>
<td>12</td>
<td>60.1094</td>
</tr>
<tr>
<td>Exponential chain</td>
<td>100</td>
<td>98</td>
<td>14</td>
<td>20</td>
<td>168.8594</td>
</tr>
<tr>
<td>Two Exponential node chains</td>
<td>10</td>
<td>5</td>
<td>-</td>
<td>3</td>
<td>53.7344</td>
</tr>
<tr>
<td>Two Exponential node chains</td>
<td>19</td>
<td>8</td>
<td>-</td>
<td>4</td>
<td>112.0625</td>
</tr>
<tr>
<td>Two Exponential node chains</td>
<td>61</td>
<td>18</td>
<td>-</td>
<td>5</td>
<td>578.2188</td>
</tr>
</tbody>
</table>

References


