Yugoslav Journal of Operations Research
23 (2013), Number 1, 43-57
DOI: 10.2298/YJOR110608018K

ELECTROMAGNETISM METAHEURISTIC ALGORITHM FOR SOLVING THE STRONG MINIMUM ENERGY TOPOLOGY PROBLEM

Aleksandar KARTELJ

Faculty of Mathematics,
University of Belgrade, Serbia
kartelj@matf.bg.ac.rs

Received: June 2011 / Accepted: August 2012

Abstract: In this paper electromagnetism (EM) metaheuristic is used for solving the NP-hard strong minimum energy topology problem (SMETP). Objective function is adapted to the problem so that it effectively prevents infeasible solutions. Proposed EM algorithm uses efficient local search to speed up overall running time. This approach is tested on two sets of randomly generated symmetric and asymmetric instances. EM reaches all known optimal solutions for these instances. The solutions are obtained in a reasonable running time even for the problem instances of higher dimensions.

Keywords: Electromagnetism, Metaheuristic, The strong minimum energy topology, Wireless networks.
MSC*:* 90C59, 05C40, 68M10.

1. INTRODUCTION

1.1. Wireless sensor networks

Wireless sensor network (WSN) can be consisted of geographically distributed autonomous sensors which cooperatively measure some phenomena like temperature, intensity of sound, vibrations, pressure etc. (for example seismic network instruments). Evolution of such networks was motivated by development of military devices for battle field surveillance, and today they are also used for different civil and industry matters, transport and pollution control (for other usages see [1]). The basic unit of the network is the wireless sensor, which is usually equipped with a measurement instrument, the radio used for communication with the rest of the network, a little programmable micro-controller and a battery.

One of the problems in WSN is to reduce the amount of energy used for power supply. One way to address this problem is to minimize costs for each sensor by cutting the cost of access and activation. Another way is to address the problem globally and try to cut the costs by choosing optimal topology of network. This way, communication between sensors will be the cheapest implying long battery life. In this paper, mathematical formulation of SMETP problem is presented and EM is used to solve it.

1.2. Problem formulation

The goal is to assign some quantity of energy to each sensor in a network so that the network becomes strongly connected and the total energy used in the whole network is minimized. More formally, for a given set of sensors distributed on the plane, certain amount of energy should be assigned to each sensor so that there is at least one directed path between every ordered pair of sensor vertices, and the total amount of energy is minimized [2]. Transmission energy between vertices (sensors), indexed with i and j, is usually defined as:

$$f_{i,j} = f(d_{i,j}) = t_j (d_{i,j})^{\alpha} . \tag{1}$$

where:

- $d_{i,j}$ is a measure of distance between vertices,
- t_j is a sensitivity threshold of sensor j,
- α is a constant related to the loss of signal energy.

Sensitivity threshold t_j of sensor j is the value of signal necessary to be detected by sensor i. In practice, sensitivity threshold for all sensors is usually equal, so its value is normalized to 1. The consequence of this normalization is that the problem becomes symmetric (in this paper asymmetric case is also analyzed). Constant α defines how fast the signal loses its strength, and its usual values are 2 or 4 (see [5], [6]). If z_i represents energy assigned to vertex i, then for every vertex j for which $f_{i,j} \le z_i$ is true, we say that it belongs to the signal area of vertex i, so it is possible to send signal from vertex i to vertex j. For a given complete digraph $D = (V, A)$, where V is a set of vertices (sensors), and A is a set of arcs (possible communication lines), graph formulation of the problem is as follows:

Definition 1. SMETP. *For a given complete digraph $D = (V, A)$ and its corresponding energy function $f_{i,j}$, assign battery energy level to each vertex, so that subgraph*

$D'=(V,A')$, *where* $A'=\{f_{i,j} \mid f_{i,j} \le z_i, i, j \in V\}$ *is strongly connected and total amount of energy* $\sum_{i=1}^{n} z_i$ *is minimized.*

Although this definition is straightforward, it is not practical, so slightly different problem will be introduced. Strongly connected spanning subgraph problem (SCSSP) is given by the following definition:

Definition 2. SCSSP. *For a given complete digraph* $D=(V,A)$ *and its corresponding energy function* $f_{i,j}$, *find minimum spanning subgraph* $D'=(V,A')$, *such that* $\sum_{i\in V} \max\{f_{i,j} \mid (i, j) \in A'\}$ *is minimized.*

It has been shown in [2] that SMETP can be easily reduced to SCSSP in polynomial time by assigning $\max\{f_{i,j} \mid j \in A'\}$ to z_i for each vertex.

1.3. Previous results

In [4], this problem appears under the name broadcast strong connectivity augmentation problem. Authors show that SMETP is NP hard and, assuming symmetric data, any minimum spanning tree (MST) gives at most two times worse solution. In [7], assuming symmetric data, approximation algorithm for solving SMETP in ad hoc networks is presented. In [6], authors presented the results obtained when relationship between the energy consumption of an optimal solution and the approximate solution of any spanning tree is taken into account. Two polynomial time approximation heuristics are proposed: one based on MST, and the other, called MST-reduced, based on the minimum incremental power tree. MST has the lexicographic property such that the longest edge in the tree is the minimum among all spanning trees. Thus, any MST solution minimizes the maximum energy used by any sensor node. Computational experiments suggest that MST-reduced tree provides an average improvement of about 4 percent over MST solution. In [11], the proposed memetic algorithm gave a better solution than the approximation of MST algorithms. In [2], different integer linear programming (ILP) formulations are presented, and Branch and cut algorithm, which uses solutions to these ILP formulations sub problems in every vertex of B&C search tree, is then applied. At the end, cutting non perspective tree parts, based on these solutions, is performed.

Similar problems are also studied in [8] and [10]. The authors solved the problem of minimizing the total power in the network under the constraint that every sensor should directly or indirectly communicate with the master node. The objective, again, is to minimize the total powers assigned to sensor nodes. SMETP differs from these two problems in a sense that it requires all-to-all communication between sensor nodes instead of communication from all sensor nodes to the master node. In [5], the problem of assigning transmitting power to each sensor, so that the induced topology contains only bidirectional links strongly connected, is considered.

2. INTEGER LINEAR PROGRAMMING FORMULATION

In this section integer linear programming formulation of SCSSP introduced in [2] is presented. Let a set X of binary variables $x_{i,j}$ be given such that $x_{i,j}=1$ if the arc (i,j) is selected to be the part of a spanning subgraph (Definition 2), otherwise $x_{i,j}=0$. Let z_i be the weight assigned to vertex i, which corresponds to energy assigned to vertex i. Furthermore, for a given set of vertices S, such that $S \neq V$ and $S \neq \emptyset$, let $(S,\overline{S})$ be the set of all arcs (i,j) that connect vertices from set S with set $\overline{S} = V \setminus S$. Then, integer linear programming formulation of the problem is as follows:

$$\min \sum_{i=1}^{|V|} z_i \tag{2}$$

subject to

$$z_i \geq w_{i,j} x_{i,j}, \qquad \textit{for every } (i,j) \in A, \ \textit{for every } i \in V \tag{3}$$

$$\sum_{(i,j)\in(S,\overline{S})} x_{i,j} \geq 1, \qquad \textit{for every } S \subset V, S \neq V, S \neq \emptyset \tag{4}$$

$$x_{i,j} \in \{0,1\}, \qquad \textit{for every } (i,j) \in A \tag{5}$$

$$z_{i,j} \geq 0, \qquad \textit{for every } i \in V \tag{6}$$

Objective function (2) represents the sum of weights of all vertices. Constraints (3) state that the weight of each vertex has to be greater or equal to the cost of each outgoing arc. This set of constraints ensures that each sensor has a sufficient amount of energy to send a signal to each of the neighboring sensors. Constraints (4) ensure that for each two distinct sets partitioning of the starting set of vertices exists a connection between at least two vertices, such that the first belongs to the first partition, and the second to the second partition. Following the transitivity rule, a directed path between any two vertices exist, which means that the graph is strongly connected. Constraints (5) state that $x_{i,j}$ is a binary variable, and constraints (6) insure that a negative value of energy for any vertex is impossible.

3. EM FOR SMETP

Electromagnetism as an optimization heuristic was proposed in [3]. This method can solve nonlinear optimization problems (the details about convergence can be found in the mentioned paper). As EM is a population based algorithm, in the following text each member p_k, $k=1..m$ of that population will be referred as an EM point (or solution point), and the population itself will be referred as a set of points (or solution set).

Each EM point in the set of points has its associated charge, which is calculated as a function of its and other point's objective functions. Every point has an impact on the others through charge, and its exact value is given by Coulomb's Law. This means that power of connection between two points will be proportional to the product of charges and reciprocal to the distance between the points. The points with a higher charge will attract other points more strongly. Besides, the best EM point will stay unchanged. Proposed EM program for solving SMETP is given in the following pseudo-code:

```
EM for SMETP
1. data input and initialization
  while (iteration less than max. iteration) do
      foreach (point pk in solution set)
           2. calculate objective value of pk and repair the
              solution if needed
           3. perform local search to improve it
           4. scale the solution
      endforeach
      5. calculate charges and forces
      6. apply forces
      if (the same solution appeared max. number of times)
           7. stop
      endif
  endwhile
```

In the initialization part, EM points are created, and then each coordinate of every EM point is randomly selected from [0,1].

3.1. Objective function

According to (2) and (3), if all variables $x_{i,j}$ are fixed, for obtaining variables z_i, it is sufficient to assign:

$$z_i = \max_{j:(i,j)\in A} f_{i,j} x_{i,j} \tag{7}$$

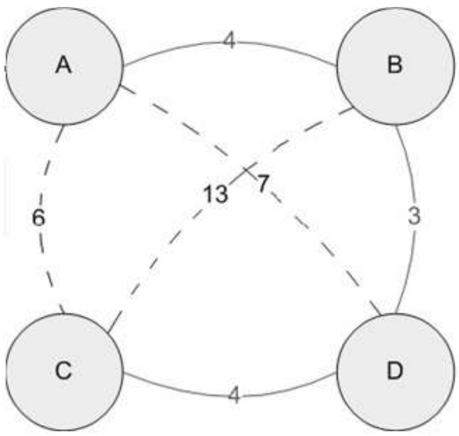
Therefore, EM point p_k represents $|A|$-dimensional vector of real value coordinates, taking values from the interval [0,1]. Since $x_{i,j}$ is a binary variable, its value is obtained by rounding the value of the corresponding *(i,j)* coordinate of p_k. These real values are mapped to binary solution vector by using threshold value δ, which is here set to 0.5.

$$x_{i,j} = \begin{cases} 1, & p_{k,(i,j)} \geq \delta \\ 0, & p_{k,(i,j)} < \delta \end{cases} \tag{8}$$

For obtained values $x_{i,j}$, it is necessary to check strong connectivity. In this implementation, modification of Tarjan's algorithm [9] is used. It is a variant of depth first search algorithm, with a following adjustment: it begins search from a randomly selected root vertex. In graph G_2 (Example 2), the first selected root vertex is C. Starting from this vertex, the first strongly connected component consisting of vertices $\{C, D, E\}$ is found. Afterward, vertex *A* is randomly selected as a new root. Random selection of root elements preserves diversity of repaired solutions.

Example 1. Let $G_1 = (V_1, E_1)$ be a graph with four vertices $V_1 = \{A, B, C, D\}$ and twelve possible edges between them $E_1 = \{AB, AC, AD, BA, BC, BD, CA, CB, CD, DA, DB, DC\}$. For the sake of readability and without loss of generality, undirected graph is considered (symmetric case) instead of directed graph. EM point represented with real vector bellow is then mapped to the following binary solution vector, and corresponding solution subgraph is given in Figure 1.

[0.90, 0.45, 0.13, 0.67, 0.51] → [1, 0, 0, 0, 1, 1]



1	AB
0	AC
0	AD
0	BC
1	BD
1	CD

Figure 1: Corresponding graph for given solution point

To each vertex, the least amount of energy z_i is assigned, which is large enough to send signals over the edges (communication lines) incidental to it. In this example, z_A is assigned the value 4, because it is the selected edge with largest weight incidental to vertex *A*, similarly for other vertices $z_A = z_B = z_C = z_D = 4$, so the total network energy is $z = 16$.

A thing to be emphasized here is that a graph has to be strongly connected, that is true in this example, and if it is not the case, than a repair procedure should be performed. Outline of this procedure is presented in the following example.

Example 2. Let's consider graph $G_2 = (V_2, E_2)$ with five vertices $V_2 = \{A, B, C, D, E\}$. This graph and its corresponding binary solution vector are depicted in Figure 2. After performing strong connectivity checking algorithm, two strongly connected components can be noticed. The first one is consisted of vertices $\{A, B\}$, and the second contains vertices $\{C, D, E\}$. Obviously, this type of topology is not allowed, so necessary repair steps are taken. During the execution of strong connectivity checking algorithm, roots of strong connectivity components are saved. The repair procedure is based on adding necessary edges between root vertices, such that the new topology becomes strongly connected. In this example undirected graph is repaired by adding the edge AC. Similarly, directed graphs are supplemented by adding arcs in both directions between appropriate root vertices.

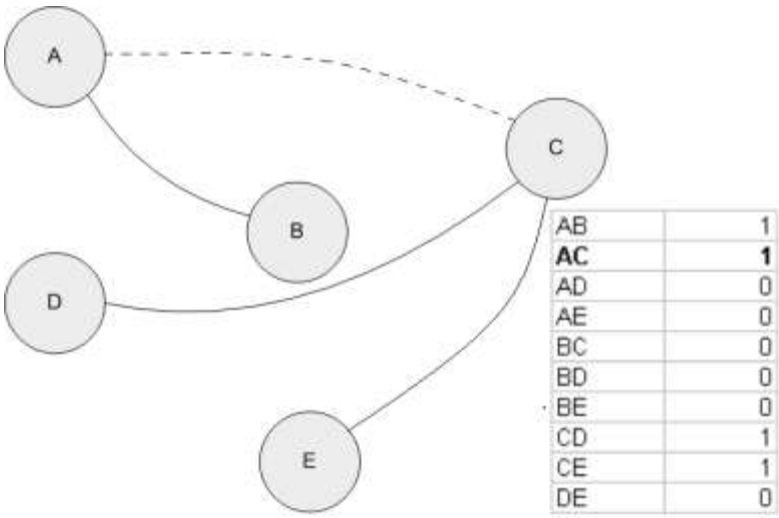


Figure 2: Graph connectivity repair

3.2. Local search

Let (u,v) be the arc, such that it belongs to the resulting subgraph $D = (V, A')$. From the condition that the resulting subgraph is strongly connected, the removal of arc (u,v) could cause that this property does not hold. Two arcs, (u,r) and (r,v), should be found such that the sum of their costs is less then the cost of the arc (u,v). When the pair of arcs that fulfill this criterion is found, (u,v) should be replaced by (u,r) and (r,v). From the condition that the resulting subgraph is strongly connected before the removal of the arc (u,v), there are two possible outcomes after this removal: the number of strongly connected components will either stay the same or will increase. The first outcome means that for each pair of vertices, even for (u,v), there is a path which does not include arc (u,v); so, introducing (u,r) and (r,v) will not make any difference regarding strong connectivity property. In the second case, this property will be satisfied. This is due to the fact that the arbitrary pair of vertices include arc (u,v) in its path, and will be connected with the path which now contains arcs (u,r) and (r,v). Also, the objective function will be improved; higher direct communication cost will be replaced by two lower, so the average communication line costs will decrease. In the long run, this will reduce the required battery level. Pseudo-code of the LS is as follows:

```
SMETP local search

1. select a random index i for circular traversal of A′
for each (arc a in A′ starting from A′[i])

      2. select a random index j for circular traversal of
      set of arc a neighbors denoted by Na

      foreach (n in Na starting from Na[j])

           3. select a random index k for circular
           traversal of set of arc n neighbors denoted by
           NNa

           foreach (nn in NNa starting from NNa[k])

               if( f(n)+f(nn)≤ f(a))

                    4. do replacement

                    5. go to 6

               endif

           endforeach

      endforeach

  endforeach

  6. update state
```

3.3. Solution scaling

In this section a scaling procedure is introduced, which influences intensification and diversification balancing. The procedure of scaling is performed after local search, and its main goal is to transform the vector in such a manner that intensification of search is increased. Scale factor r_i is obtained by decreasing each EM coordinate value with lower bound, and after that dividing it with the difference between upper and lower bound, in other words by normalizing it to the interval [*lb*,*ub*].

```
Solution scaling
foreach(arc (i,j) in A)
        r_k,(i,j) = ( p_k,(i,j) -lb)/(ub-lb)
        if(x_i,j=1)
                p'_k,(i,j) = (1- r_k,(i,j))*δ + r_k,(i,j)*ub
        else
                p'_k,(i,j) = r_k,(i,j)*δ + (1-r_k,(i,j))*lb
        endif
endforeach
```

Example 3. Consider the vector of real values given in Example 1. The effect of solution scaling is given as:

[0.90, 0.45, 0.13, 0.67, 0.51] → [0.95, 0.225, 0.065, 0.835, 0.755].

Here, the following values of parameters are assumed: $lb = 0,\ ub = 1,\ \delta = 0.5$. The value of the first coordinate ($0.90 > \delta$) implies that its corresponding arc belongs to solution subgraph, so calculation is as follows:

$$r_{k,(1,2)} = (0.9-0)/(1-0) = 0.9$$
$$p'_{k,(1,2)} = (1-0.9)*0.5 + 0.9*1 = 0.95$$

3.4. Charges and forces

EM points are being evaluated by calculating their charges q_k, given the following formula:

$$q_k = \exp\left(-|A|\frac{f(p_k)-f(p_{best})}{\sum_{l=1}^{m}(f(p_l)-f(p_{best}))}\right). \tag{9}$$

where $|A|$ is dimension space and p_{best} is the best EM point so far. Finally, total force that these charges produce is applied. The resulting force F_l on point l is the sum of force vectors induced by all other neighbor points on point l:

$$F_l = \sum_{k=1,k\neq l}^{m} \left\{ \begin{array}{l} (p_l - p_k)\left(\dfrac{q_l q_k}{\| p_k - p_l \|^2}\right), if\ f(p_k) < f(p_l) \\ (p_k - p_l)\left(\dfrac{q_l q_k}{\| p_k - p_l \|^2}\right), if\ f(p_k) \geq f(p_l) \end{array} \right\}. \qquad (10)$$

Where $\| p_k - p_l \|$ is Euclidian distance between vector p_k and p_l:

$$\| p_k - p_l \| = \sqrt{\sum_{(i,j)\in A} (p_{k,(i,j)} - p_{l,(i,j)})^2}\ . \qquad (11)$$

Calculated value of F_l is normalized, so it represents a direction in which a point is going to move. New location of the point will depend on its objective value, current location in space, lower and upper bound of domain, but also on stochastic element λ, which quantifies the size of movement. In this implementation, value of λ is set to random number between 0 and $ln(|A|)$. Pseudo-code of this procedure is given bellow:

```
Solution update
foreach (point p_k in solution set)
      λ = a random number between 0 and ln(|A|)
      foreach (arc (i,j) in A)
            if (F_k,(i,j)>0)
                  p'_k,(i,j) = p_k,(i,j) + λ* F_k,(i,j) *(ub- p_k,(i,j)) /norm(p_k)
            else
                  p'_k,(i,j) = p_k,(i,j) + λ* F_k,(i,j) *( p_k,(i,j) -lb) /norm(p_k)
            endif
      endforeach
endforeach
```

In this pseudo-code, $norm(p_k)$ represents the Euclidian norm of point p_k, $p_{k,(i,j)}$ and $p'_{k,(i,j)}$ are old and new (i,j) coordinate of point p_k, while $F_{k,(i,j)}$ is (i,j) coordinate of the force applied on point p_k.

4. EXPERIMENTAL RESULTS

EM based algorithm for solving SMETP is implemented in C programming language, and compiled in Visual Studio 2010. ILP formulation was used to perform testing for smaller instances using CPLEX solver, version 12.1. Computational tests were performed on PC, with Intel 2.0 GHz processor, under Windows 7 operative system.

Computational times are measured in seconds. EM has the following parameters: number of solution points in solution set (this implementation uses 70), maximum number of iterations (20000), maximum number of iterations yielding the same result (5000), threshold between lower and upper bound (0.5) and random seed. When dealing with NP hard problems like SMETP, it is not possible to verify optimality of solution, so CPLEX was run on smaller instances to check the EM solution quality.

4.1. Instances

Two sets of random weighted graphs are generated, where all instances are complete digraphs. The first set is symmetric, thus the transmission energy of the signal in both ways is the same, which is the consequence of normalized threshold sensitivity assumption mentioned earlier in the paper. For both sets of instances, vertices are distributed randomly as integer vectors in [0,10000]x[0,10000] area, and arc weight between two vertices is given by formula $(d_{i,j})^2$, as it was shown in Section 1.2. In case of asymmetric instances, weight between vertices i and j is calculated as $t_j(d_{i,j})^2$ where t_j is in the interval [1, 2].

4.2. Results

In Tables 1-2, the first column is the name of the instance, the second represents dimension, the third is optimal value obtained by CPLEX value in case when it finished its work, t_{cplex} is the corresponding CPLEX running time. Last three columns describe EM results: em_{best} is the best solution EM produced, t_{em} is the time needed to obtain the best EM solution, and the last column t_{emtot} represents the total time needed to reach finishing criteria. EM cannot prove optimality, so it waits for the finishing criterion to exit execution. The difference $t_{em} - t_{emtot}$, represents the time for which algorithm finishes execution after the best solution is found. Mark „*" denotes that the time limit of 2 hours was exceeded. Table 3 is organized similarly as Table 1 and Table 2, but the instances in Table 3 were not reachable by CPLEX, so optimal solution value and CPLEX running time are omitted.

Table 1: Results for small symmetric instances

instance	$\|A\|$	sol_{cplex}	t_{cplex}	em_{best}	t_{em}	t_{emtot}
sym101	90	2185	9.50	*opt*	58.14	86.14
sym102	90	2067	8.07	*opt*	49.91	80.61
sym103	90	2808	5.72	*opt*	57.96	93.28
sym104	90	2576	8.90	*opt*	50.79	82.78
sym105	90	2376	5.67	*opt*	51.90	80.54
sym111	110	2939	37,88	*opt*	100,9	244,99
sym112	110	2521	64,69	*opt*	92,1	238,19
sym113	110	2405	58,17	*opt*	90,13	237,08
sym114	110	2370	36,82	*opt*	113,93	257,84
sym115	110	2397	125,88	*opt*	118,82	263,17
sym121	132	2566	1550,49	*opt*	180,77	351,37
sym122	132	3177	945,43	*opt*	113,11	279,81
sym123	132	3101	560,4	*opt*	133,76	298,41
sym124	132	2804	820,09	*opt*	148,79	320,27
sym125	132	2892	227,06	*opt*	136,25	300,44
sym131	156	3048	*	*3047*	177,34	381,29
sym132	156	3275	4366,46	*opt*	208,03	402,64
sym133	156	2755	*	*2755*	185,42	385,71
sym134	156	2897	*	*2897*	201,45	397,52
sym135	156	2771	*	*2771*	185,6	384,07

Table 2: Results for small asymmetric instances

instance	$\|A\|$	sol_{cplex}	t_{cplex}	em_{best}	t_{em}	t_{emtot}
asym101	90	4084	11.28	*opt*	54.08	87.32
asym102	90	3576	8.36	*opt*	56.23	82.28
asym103	90	3340	5.69	*opt*	51.36	83.07
asym104	90	3435	5.45	*opt*	48.51	82.10
asym105	90	3641	3.67	*opt*	55.63	88.42
asym111	110	4110	8,26	*opt*	63,89	207,56
asym112	110	3722	39,51	*opt*	119,79	263,79
asym113	110	3547	31,22	*opt*	105,56	251,25
asym114	110	3178	10,66	*opt*	104,46	246,99
asym115	110	3251	7,66	*opt*	87,83	232,79
asym121	132	3761	313,28	*opt*	127,42	292,75
asym122	132	4476	152,47	*opt*	145,39	317,13
asym123	132	3712	184,64	*opt*	150,06	322,27
asym124	132	3501	451,66	*opt*	140,61	312,56
asym125	132	4959	2516,23	*opt*	144,43	315,64
asym131	156	4902	3338,9	*opt*	193,61	388,43
asym132	156	3720	1631,13	*opt*	164,52	359,5
asym133	156	4354	2913,39	*opt*	283,32	476,99
asym134	156	3937	1837,25	*opt*	140,44	335,15
asym135	156	4859	137,24	*opt*	167,22	382,75

In Table 3 EM results for symmetric and asymmetric instances with dimensions 210 to 2450 are shown. Solutions for these instances were not reachable by CPLEX, so optimal solution is not known.

Table 3: Results for large symmetric and asymmetric instances

instance	$\|A\|$	em_{best}	t_{em}	t_{emtot}	instance	em_{best}	t_{em}	t_{emtot}
sym15	210	4499	132.25	211.00	asym15	5513	178.43	202.22
sym20	380	5454	166.06	313.97	asym20	3038	296.31	378.90
sym25	600	5632	494.72	574.86	asym25	3422	350.41	580.12
sym30	870	9257	815.36	842.42	asym30	4995	1058.36	1064.03
sym35	1190	13744	1163.42	1165.05	asym35	9005	1131.28	1147.67
sym40	1560	18598	1508.11	1520.06	asym40	12036	1514.74	1525.52
sym45	1980	26379	1905.67	1934.11	asym45	14401	1935.31	1940.23
sym50	2450	27962	2343.40	2372.69	asym50	18118	2350.55	2373.91

From Table 1 and Table 2, it is evident that EM reached all known optimal solutions sym101-sym125, sym132, asym101-asym135. As can be seen from the tables, running times are relatively short. For small instances, total running time is up to 477 seconds. Also, it can be seen that optimal solutions were always reached in shorter running time, but finishing criteria were not satisfied. For example on asym133 instance total running time was 476.99 seconds, but the optimal solution, whose value is 4354, was reached only after 283.32 seconds. This is quite shorter than CPLEX, whose running time was 2913.39 seconds. Although CPLEX is general solver, so running time of EM can not be directly comparable with the running time of CPLEX, experimental results show that EM is quite competitive. Experimental results on large instances presented in Table 3 are even more promising. Running time was attainable, i.e. for all large instances it was less than 40 minutes.

5. CONCLUSIONS

In this paper electromagnetism metaheuristic algorithm for solving the problem of strong minimum energy topology is presented. Appropriate objective function mapping real valued EM points to binary SMETP solution is implemented. Proposed objective function effectively prevents infeasible solutions. The scaling of real vectors to obtained integer solutions directs the search towards promising search regions. The experimental results show that EM matches with all known optimal solutions of tested instances. The running times were reasonable even for large instances.

One possible direction for the further work is parallelization of the presented algorithm so it can be tested on multiprocessor system. Also, hybridization of the algorithm with other exact or heuristic methods is possible.

REFERENCES

[1] Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E., “Wireless sensor networks: A survey”, *Computer Networks,* 38 (2001) 393-422.

[2] Aneja Y.P., R. Chandrasekaran, Xiangyong Li, and K.P.K. Nair, “A branch-and-cut algorithm for the strong minimum energy topology in wireless sensor networks”, *European Journal of Operational Research,* 204 (2009) 604-612.

[3] Birbil, S.I. and Fang, S.C, “An electromagnetism-like mechanism for global optimization”. *Journal of Global Optimization,* 25 (2003) 263-282.

[4] Chen, W.-T., and Huang, N.-F., “The strongly connecting problem on multihop packet radio networks”. *IEEE Transaction on Communications,* 37 (1989) 293-295.

[5] Cheng, X., Narahari, B., Simha, R., Cheng, X., and Liu, D., “Strong minimum energy topology in wireless sensor networks: NP-completeness and heuristics”. *IEEE Transactions on Mobile Computing,* 2 (2003) 248-256.

[6] Cheng, M.X., Cardei, M., Sun, J., Cheng, X., Wang, L., Xu, Y., and Du, D.Z., “Topology control of ad hoc wireless networks for energy efficiency”. *IEEE Transactions on Computers,* 53 (2004) 1629-1635.

[7] Lloyd, E., Liu, R., Marathe, M., Ramanathan, R., and Ravi, S., “Algorithmic aspects of topology control problems for ad hoc networks”, in: *Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing - MobiHoc2002,* Lausanne, Switzerland, June 9-11, 2002, 123-134.

[8] Rodoplu, V., and Meng, T.H., "Minimum energy mobile wireless networks". *IEEE Journal on Selected Areas in Communications,* 17 (1999) 1333–1344.

[9] Tarjan Robert, "Depth-first search and linear graph algorithms". *SIAM Journal on Computing*, 1 (1972) 146-160.

[10] Wattenhofer, R., Li, L., Bahl, P., and Wang, Y.-M., "Distributed topology control for power efficient operation in multihop wireless ad hoc networks", in: *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies - INFOCOM'2001,* Anchorage, Alaska, April 22–26, 2001, 1388–1397.

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