

IMPROVING THE SPEED OF DYNAMIC CLUSTER FORMATION IN MANET VIA SIMULATED ANNEALING

K. Manousakis* and J.S. Baras
Electrical and Computer Engineering Department
and the Institute for Systems Research
University of Maryland College Park
College Park, MD 20742

A.J. McAuley and R. Morera
Telcordia Technologies Inc.
One Telcordia Drive
Piscataway, NJ 08854

ABSTRACT

Future military systems, such as FCS, require a robust and flexible network that supports thousands of ad hoc nodes. Therefore, networking protocols for MANETs must be made to scale. The use of hierarchy is a powerful general solution to the scaling problem. We have previously proposed methods based on Simulated Annealing (SA) to optimize hierarchy in MANETs. The challenge, however, is to improve the slow convergence time of SA, so it can be used in dynamic environments, without penalizing optimality. In previous work the importance of parameters such as cooling schedule, state transition probabilities and convergence condition are investigated. This paper proposes a new approach to decrease SA convergence time. SA is an optimization technique based on an iterative process that takes an initial solution, or map, to start the process. In this paper we analyze the effect that this initial solution has on the SA convergence time as a function of the network size. We believe that the combined modifications to SA can speed the optimization process to the point that it can quickly generate very efficient clustering solutions in large dynamic networks.

1. INTRODUCTION

In recent years Mobile Ad Hoc Networks (MANETs) have become very popular, due to their infrastructure-less characteristics. Their importance in the military world has been highlighted from many researchers and military planners. However, there are still many unresolved problems that make them inefficient in scenarios such as the large scale ad hoc networks envisioned in Future Battlefield Networks. Then, networking protocols (e.g., routing, security and QoS) must be made to scale.

The use of hierarchy is a powerful general solution to the scaling problem, since it allows networking protocols to operate on a limited number of nodes, as opposed to the entire network. Hierarchy also provides other important benefits, such as simplifying network management and providing more efficient support for heterogeneity.

In the literature, there are many research proposals for dynamically creating and maintaining an optimal hierarchy in large dynamic networks (Lin and Gerla, 1997; Baker et al., 1984; Chatterjee et al., 2002; Basagni, 1999). While local maintenance algorithms are essential to provide fast robust performance, the use of a global optimization has been shown to be critical on providing good overall clustering. In particular, Simulated Annealing (SA) can optimize the network for a wide variety of metrics simultaneously (Manousakis et al., 2004; Manousakis and Baras, 2004). In (Manousakis et al., 2004) SA was optimized for the specific application of clustering by a) modifying the termination condition of the algorithm, b) selecting faster cooling schedule and c) modifying the state transition probabilities in accordance to the cost function being optimized. These techniques produced an order of magnitude improvement in performance compared to standard SA.

Despite the improved performance, the application of SA is still limited to clustering hundreds of nodes. Other techniques (e.g., min-cut) are available for larger networks, but they are unable to solve the required complex multi-metric, multi-layer optimizations. This paper proposes new techniques to provide further significant reduction in SA convergence time that will allow applying SA in networks with thousands of nodes.

Our approach is based on providing a clustering map “better” than a random map as the initial cluster configuration for SA to start the optimization process. Although this is a well founded intuitive solution, it has not been shown how much reduction in convergence time can be achieved and how much loss in optimality results for typical clustering cost functions. Even though SA wanders randomly around the surface of feasible

Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance (CTA) Program, Cooperative Agreement DAAD19-2-01-0011. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 00 DEC 2004		2. REPORT TYPE N/A		3. DATES COVERED -	
4. TITLE AND SUBTITLE Improving The Speed Of Dynamic Cluster Formation In Manet Via Simulated Annealing				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Electrical and Computer Engineering Department and the Institute for Systems Research University of Maryland College Park College Park, MD 20742; Telcordia Technologies Inc. One Telcordia Drive Piscataway, NJ 08854				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release, distribution unlimited					
13. SUPPLEMENTARY NOTES See also ADM001736, Proceedings for the Army Science Conference (24th) Held on 29 November - 2 December 2005 in Orlando, Florida. , The original document contains color images.					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 8	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

solutions, if it starts from a solution that is closer to optimal it is more probable to reach the optimal faster and stay there for the appropriate number of iterations until final convergence.

Any optimal solution must work within the constraints that any pair of nodes in the same cluster can reach each other using only links and nodes that are internal to the cluster. We call clusters having this pathwise property *feasible clusters*; they fall within the category of topological clusters as defined below:

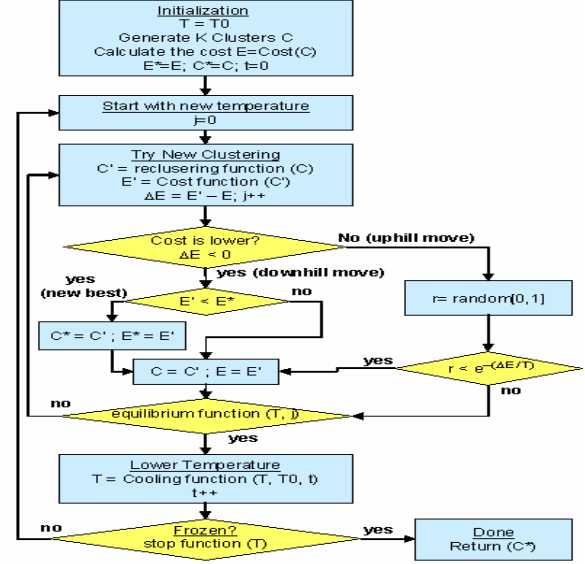
Definition (Topological Cluster): A cluster consisting of the set $\mathbb{S}$ of nodes is called topological if $\forall node_i, node_j \in \mathbb{S}$ and $i \neq j$, there is always a path P_{ij} from $node_i$ to $node_j$ such that $\forall node_k \notin \mathbb{S}$ holds that $node_k \notin P_{ij}$. All the members of a cluster can communicate between them without the need to use inter-cluster links, which are links that involve non-member nodes.

The initial cluster map that is fed into the SA must be feasible. In (Manousakis et al., 2004), the initial configuration map is generated by randomly assigning nodes to clusters, with the constraint that the generated clusters are feasible. We propose here two ways of generating better initial clusters either by applying some fast heuristic related to the optimization metric of SA, or, when SA runs periodically, by using the existing or previously optimized clustering (so we do not start from scratch each time).

The use of a modified SA solution is particularly attractive in dynamic networks which must be re-optimized relatively often. Obviously, due to the mobility of nodes, clusters may become infeasible. Therefore we must force feasibility before a previously generated cluster configuration can be input to SA. Alternatively, we can directly use the current solution produced by local maintenance algorithms. Different approaches can be used (Manousakis et al., 2004b) that can have different effects on the rate of degradation, but all maintain feasible solutions that can be fed in directly into SA.

2. SIMULATED ANNEALING

Simulated annealing (SA) has been widely used for tackling different combinatorial optimization problems (Kirkpatrick et al., 1983). The process of obtaining the optimum configuration is similar to that followed in a physical annealing schedule. In SA, however, the temperature is merely used as a control parameter and does not have any physical meaning.



Inputs	Examples
Equilibrium function	Constant ($t = 5000$), "Stop repeats" (function of
Cost function	$\sum_{i=1}^K \text{Diameter}(C_i)$
Cooling function	Geometric or logarithmic
Stop function	Minimum temperature (e.g., $T=0.1$), "Stop repeat" criteria
Reclustering function	Random move of one node
T_0	Initial Temperature
K	Number of clusters

Variable	Definition
T	Current Temperature
C	Current Cluster map (e.g., $\{1,2\},\{3,6\},\{4,5\}$)
C'	New cluster map to test (e.g., $\{1,2\},\{3,5,6\},\{4\}$)
C^*	Champion cluster map (e.g., $\{1,6\},\{3,5\},\{2,4\}$)
E	Current cost
E'	Cost of new cluster map
E^*	Champion cost
j	Inner loop counter
t	Outer loop counter

Figure 1. Simulated Annealing algorithm for network partitioning

Figure 1 highlights the general steps in the algorithm. The objective of the algorithm is to obtain the K cluster network partition configuration, C^* , that optimizes a particular cost function. The process starts with an initial temperature value, T_0 , which is iteratively decreased by the cooling function until the system is frozen (as decided by the stop function). For each temperature, the SA algorithm takes the current champion configuration C^* and applies the recursive function to obtain a new configuration C' and evaluates its cost, E' . If E' is lower than the cost of the current E^* , C' and E' replace C^* and E^* . Also, SA randomly accepts a new configuration C' even though E' is greater than E^* to avoid local minima. In the latter case C' and E' replace C^* and E^* respectively. One of the key characteristics of simulated annealing is that it allows uphill moves at any time and relies heavily on randomization (Johnson and McGeoch, 1997). The higher the temperature, the higher the probability of accepting a configuration that worsens E^* instead of improving it. Indeed, if the temperature is sufficiently high, SA will simply take a random walk around the solution space. The lower the temperature, the lower the probability of accepting worse configurations.

The number of iterations required to reach equilibrium is defined by the equilibrium function. The function can be a simple constant (e.g., 100) or a function of the temperature and other parameters specific to the optimization problem, such as number of network nodes.

3. COST FUNCTIONS

SA is one of many global optimization algorithms that we can utilize to obtain optimal or suboptimal clustering decisions (Manousakis et al., 2004; Manousakis and Baras, 2004). The goodness of the clustering decisions depends primarily on the cost functions and constraints provided for optimization, not on the optimization algorithms themselves. We have found the careful design and selection of cost functions is very important for the quality of clustering decisions, with respect to the imposed network objectives (e.g., minimum overhead or minimum latency). The cost functions are based on various metrics of interest that can be measured from the network. Table 1 lists ten example cost functions have been shown to meet different clustering objectives (Manousakis et al., 2004; Manousakis and Baras, 2004).

Table 1 Example Objectives and associated Cost Functions

Objective	Cost Function
Balanced Size Clusters	$J(C) = \min \left(\text{Var}(C_1 ^2, \dots, C_K ^2) \right) \quad (1)$ $J(C) = \min \sum_{i=1}^K C_i ^2 \quad (2)$
Balanced Diameter Clusters	$J(C) = \min \sum_{i=1}^K (d_{C_i}^2) \quad (3)$ $J(C) = \min \left(\text{Var}(d_{C_1}^2, d_{C_2}^2, \dots, d_{C_K}^2) \right) \quad (4)$
Balanced Size Clusters with the least number of Border Routers	$J(C) = \min \left(\text{Var}(C_1 ^2, \dots, C_K ^2) + \left(\sum_{i=1}^K BR_{C_i} \right) \right) \quad (5)$
Cluster members move in a similar direction, so we expect longer durations of stable cluster membership	$J(C) = \min \left[\sum_{z=1}^K \left(\sum_{i,j=1}^{ C_z } \theta_{r_{i,j}} \right)^2 \right] \quad (6)$ $J(C) = \min \left[\sum_{z=1}^K \text{Var}(\theta_{r_{12}}^z, \dots, \theta_{r_{ C_z -1 C_z }}^z) \right] \quad (7)$
Cluster members have similar velocity, so we expect more stable cluster membership	$J(C) = \min \left(\sum_{z=1}^K \left[\sum_{i,j=1}^{ C_z } U_{r_{i,j}}^2 \right]^2 \right) \quad (8)$

Links among cluster members have long expiration time estimates. Improves the lifetime of the generated hierarchy	$J(C) = \min \left[- \sum_{z=1}^K \left(\sum_{i,j=1}^{ C_z } I_z(LET_{ij}) \right)^2 \right] \quad (9)$
Cluster members move with similar direction and velocity, so we expect more stable cluster membership. Like (6),(7),(8) capturing more node dynamics (e.g., direction and velocity)	$J(C) = \min \left(\sum_{z=1}^K \left[\frac{\sum_{i,j=1}^{ C_z } U_{r_{i,j}}}{2 * \max(S)} \right]^2 + \sum_{i,j=1}^{ C_z } \frac{\theta_{r_{i,j}}}{180} \right) \quad (10)$

Parameter	Definition
C_i	Cluster i
$ C_i $	Size of cluster i
d_{C_i}	Diameter of cluster i
BR_{C_i}	Number of border routers of cluster i
$\theta_{r_{i,j}}$	Relative direction of nodes i,j
$U_{r_{i,j}}$	Relative Velocity of nodes i,j
LET_{ij}	Expiration Time of Link between nodes i,j
S	Scalar speed of node

4. IMPROVING THE SPEED OF SA

The most dominant characteristic of the network environments under consideration is the dynamic topology. The SA algorithm, due to its slow convergence time, has only been considered for static problems. Due to its simplicity of implementation, its ability to handle complex cost functions, and optimality of the solutions it provides, we have attempted to modify it appropriately, so that the convergence time is improved significantly. This way, convergence time is not a limitation for applying SA in dynamic networks. These improvements are not coming for free, but affect the optimality of the clustering solutions provided by SA. So, our task becomes twice as difficult since we have to speed up SA to the point where it can be applied in dynamic environments but also we have to be careful so that the quality of the clustering solutions remains unaffected.

In (Manousakis et al., 2004) we introduced three modifications for improving the speed of SA. In this section we will briefly review them prior to the introduction of the new modification. The three previously presented modifications are related to the following parameters of SA algorithm:

- a. Cooling Schedule
- b. Termination Condition
- c. State Transition Probabilities

A more detailed description of these modifications and their effect on the convergence time of the algorithm is given in the following paragraphs.

4.1 Cooling Schedule

The SA algorithm has been proven to convergence in probability to the global optimal under the assumption of the logarithmic cooling schedule utilization.

$$T_t = \frac{T_0}{1 + \ln t} \quad (11)$$

where,

T_0 : Initial Temperature

t : Number of iterations

Even though the logarithmic cooling schedule produces high quality clustering solutions, it also results in slow convergence times. Our objective was to select another cooling schedule, which could result in much faster convergence times but also maintain the quality of the clustering solutions. The geometric cooling schedule met our objective, since it is much faster and we obtained clustering solutions of the same quality as the logarithmic cooling schedule. The geometric cooling schedule is described from the following expression:

$$T_t = \alpha^t T_0 \quad (12)$$

where,

T_0 : Initial Temperature

t : Number of iterations

α : Constant value ($0.90 \leq \alpha \leq 0.99$)

4.2 Termination Condition

The theoretical version of the algorithm terminates when the temperature T_t reaches zero, because this is when the algorithm converges to the global optimal value. In practice the algorithm reaches the zero temperature in infinite number of iterations, so we cannot utilize this termination condition in a realizable version of SA. Obviously, the larger the number of iterations are, the better the quality of the clustering solution and the larger the running time of the algorithm (i.e., slower convergence times).

Our objective is to trade off optimality with convergence time, such that the quality is satisfying and the convergence time of the algorithm is improved. The parameter we introduced to control the termination condition is called *StopRepeats*. This value determines the number of consecutive times we have to observe the same optimal value before we stop the algorithm and accept this optimal value as the clustering solution.

In (Manousakis et al., 2004) we measured the effect of various values of *StopRepeats* on the quality of the clustering solutions. We observed that the utilization of small values $O(100)$ does not affect notably the quality of the clustering solution and at the same time improves significantly the convergence time of the algorithm.

4.3 State Transition Probabilities

The functionality of SA is based on the random generation of a new clustering map in each iteration. This clustering map is compared with the optimal at that time in order to be decided if the new clustering map takes the place of the optimal or the previous optimal is maintained.

The clustering maps that are used for the comparisons with the optimal are generated by randomly moving nodes between clusters. The generation of a new clustering map consists of three random selections:

1. Cluster from where we will move a node
2. Node from the above cluster
3. Cluster where we will move the new node

Traditionally the selection process for each one of the three elements above utilizes the uniform distribution. Our objective was to boost the ability of SA to reach a good clustering solution as fast as possible. Instead of relying on the uniform distribution for the elements 1 and 3 we decided to bond the selection probabilities on the cost function we want to optimize and study the effect that it has on the convergence time of the algorithm. This modification is expected to result in faster convergence times since it assigns higher probabilities to the potentially better clustering maps with respect to the cost function being optimized.

For a better understanding of the state transition probabilities modification, assume cost functions (1) and (2) for the generation of balanced size clusters. For this clustering objective (e.g., balanced size clusters) instead of utilizing the uniform distribution for selecting 1 and 3 to generate a new clustering map, we introduced the following state transition probabilities:

- For selecting the cluster from where we will migrate a node:

$$P(\text{"from" cluster } i) = \frac{|C_i|}{(\text{Total Number Of Nodes})}$$

- For selecting the cluster where the node will migrate to:

$$P(\text{"to" cluster } i) = \frac{(\text{Total Number Of Nodes}) - |C_i|}{(\text{Number of Clusters} - 1) \times (\text{Total Number Of Nodes})}$$

The selection of the above transition probabilities aims on the migration of nodes from larger clusters to smaller ones. By giving higher probability to this type of transitions we force the earlier generation of clustering maps with balanced size clusters. In combination with the modification related to the termination condition we force the faster convergence of the SA algorithm.

4.4 Initial Solution

The new modification we propose here is to use "better" initial solutions used to bootstrap the SA algorithm. Prior the introduction of this modification the initial solution was generated randomly subject to the constraint of topological clusters. The successful application of the modification depends on two factors:

1. The level of the improvement we get with respect to the quality of the initial solution.
2. How we can generate initial solutions that will provide us with large convergence time improvements.

In this work we will go into the details of the first factor. Due to the random nature of SA, it is not straightforward that by starting from a better than a random initial solution will provide us with any convergence time improvement at all. We will answer to the latter by quantifying the convergence time effect subject to the quality of the initial solution.

Intuitively, we would expect that the closer the value of the initial solution to the resulting optimal one, the faster the convergence time of the algorithm. The results that will be presented later provide also some indicative values of the improvement percentage.

By showing that the random character of SA algorithm quest to the optimal solution does not eliminate the effect of starting from a good initial solution, indicates that we have to provide answers for the second factor. What are these methods that can be used to determine quality initial solutions? Even though we do not deal with this problem in this paper, the approaches depend on the clustering objectives we set. Good initial solutions can be generated from heuristic methods customized to the clustering objectives (i.e., for the generation of balanced size clusters we can generate initial solutions utilizing a customized min-cut algorithm). Also, modified

optimization algorithms can be useful for the generation of quality initial solutions. Apart from the application of other methods we can also utilize a feasible, previously generated optimal solution from SA. Due to the dynamics of the MANETs environment, the clustering decisions have to undergo corrections in order to retain their optimality subject to the topology changes. In the case where we have to reapply SA, then instead of generating a new initial solution we can use as a bootstrapping clustering map the previously optimal one, under the condition that it is still feasible with respect to the new topology. The latter approach can provide quality initial solutions especially when the topology is slowly changing with respect to the reapplication frequency of SA algorithm.

In any case what we have to take into consideration when we deal with the approach will follow for the generation of initial solution, is that the combined time of the generation of the initial solution and the convergence of SA has to be smaller than the convergence time of SA when we start from a randomly selected initial solution. In other words, the following inequality has to be satisfied at all times if we want the effect of initial solution to be visible on the convergence time of the algorithm.

$$\begin{aligned} T_{nris} \leq T_{ris} &\Rightarrow t_{gnris} + t_{SA} \leq t_{gris} + t'_{SA} \Rightarrow \\ &\Rightarrow t_{gnris} - t_{gris} \leq t'_{SA} - t_{SA} \Rightarrow \Delta t_{gis} \leq \Delta t_{SA} \end{aligned}$$

where,

nris : Non-random initial solution

ris : Random initial solution

T : Complete process time

t : Partial process time

The results that will be presented in the next section can be utilized to suggest the order of the quantities on the expressions above for different quality levels of the initial solution. These results can be very helpful when we design or modify the methods for the generation of initial solutions.

5. PERFORMANCE EVALUATION

This section shows the benefits received from better initial solutions in example networks. Specifically, by bootstrapping SA with a better than a random initial clustering solution, its convergence time is improved, which is not straightforward due to its random search character. Also, we can quantify this improvement on the convergence time by looking at some indicative results related to the convergence time of the algorithm subject to various qualities of initial solutions. The following results collected for two networks of different sizes – 100 nodes

and 200 nodes. The topology of these two networks appears in the following figures.

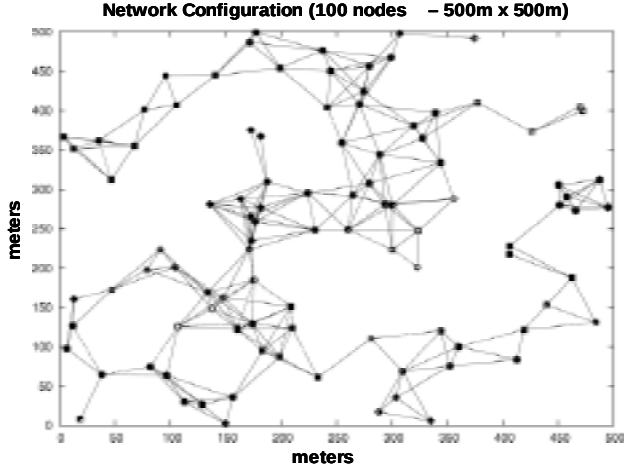


Figure 2 100 node network in an area of 500m x 500m

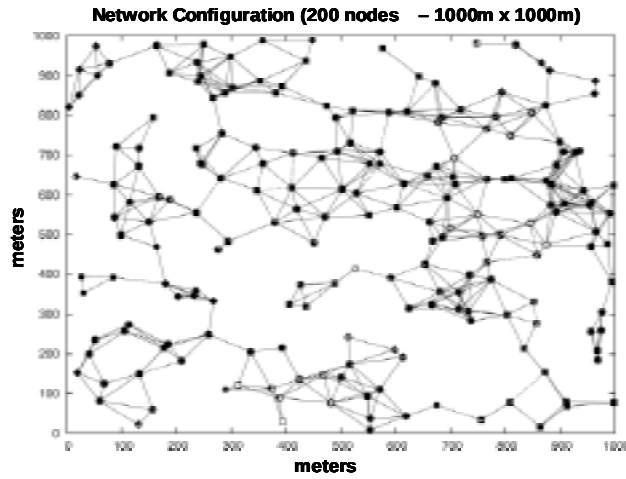


Figure 3 200 node network in an area of 1Km x 1Km

The methodology we applied for the collection of results it is based on the networks presented above and the SA clustering algorithm presented in section 2. We utilized cost function (1) for the generation of balanced size clusters. We computed the cost of a random generated solution with respect to this cost function and then we generated clustering solutions with a cost that was a fraction of the cost of this random solution.

The y-axis of the following figures represents the convergence time in seconds for various qualities of initial solutions, which are characterized from the fraction of their cost compared to the random initial solution. The x-axis is marked with the value of this fraction. The convergence time values represented from the following graphs are indicative for the cost function (1) but they provide some very useful conclusions for the proposed

modification. The following results have been averaged out from a large number of runs $O(1000)$.

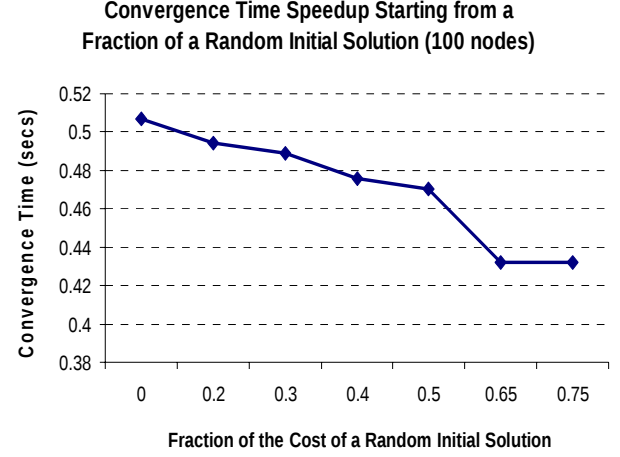


Figure 4 SA convergence time improvement with the quality of the initial solution for 100 nodes network

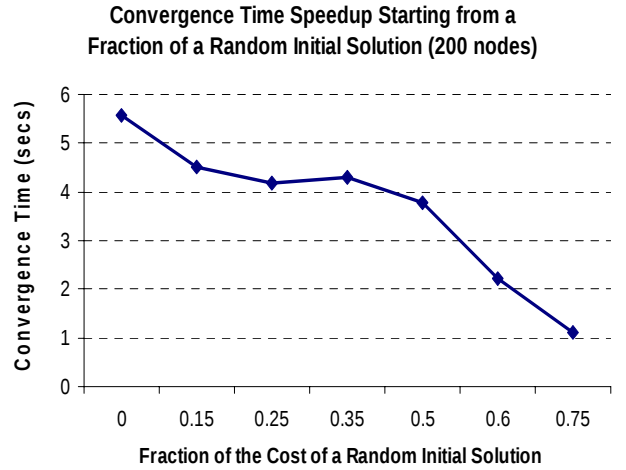


Figure 5: SA convergence time improvement with the quality of the initial solution for 200 nodes network

The most important conclusion can be drawn from the above figures is that the bootstrapping of SA algorithm with a better than a random initial solution has advantageous effect on the convergence time of the algorithm. Both curves in figures 4 and 5 respectively present a dropping tendency with the improvement of the initial solution compared to the random one. Furthermore, we can draw important conclusions also from the quantification of the convergence time improvement. By comparing the curves of figures 4 and 5 respectively, we observe that the larger the network, the larger appears to be the improvement of the convergence time. For the specific case we present, for the network of 100 nodes, the convergence time drops ~100ms as we utilize instead of a random solution, a clustering map that has a cost 75% better. For the network of 200 nodes the convergence time is improved by ~4.8s when we utilize an initial clustering

solution that is 75% better than a random one. To quantify better this decrease of convergence time we provide the percentage of improvement instead of the actual times.

Table 2: Percentage improvements of convergence time

Network size	Actual Time Improvement	Percentage of Improvement
100	0.1s	19.2%
200	4.8s	82.8%

By looking at the corresponding percentages of improvement with reference to the convergence time when SA starts from a random solution, it is obvious that the larger the network, the slower the convergence time of SA, which results in larger space of improvement. The proposed modification is sufficient to improve enough the convergence time and constitute a traditionally slow convergence algorithm, realizable even in dynamic environments like the MANETs.

We showed that better than a random initial solution results into faster convergence times for the modified SA. The remaining open issue is how we can generate these initial solutions, so that we can make the modification applicable in real world problems. As we mentioned above we can take various approaches for the design of methods that can produce quality initial solutions with respect to the clustering objectives we set. After the discussion we had in this paper, the designed methods can be characterized and compared using the fraction of the produced clustering solution cost compared to a random generated solution. This fraction in accordance with the network size can be indicative of the improvement on the convergence time as we showed from the demonstrated results.

6. CONCLUSIONS

Simulated Annealing is a well known method for the global optimization of complex cost functions. Traditionally, its slow convergence time was prohibitive for its utilization in dynamic environments and in real time problems. In this paper we overviewed previously presented modifications but also we have suggested and reviewed a newly introduced modification, which is related to the bootstrapping solution that is provided to SA algorithm during its initialization.

Due to the random search character of SA it is not straightforward that by starting from a good initial solution, this will result in faster convergence times. Here we prove this and also we quantify the improvement we get with respect to the level of quality of the initial solution (e.g., the fraction of the cost compared to a random initial solution). The results we collect regarding the proposed modification along with the improvements

we have gotten from the application of the previous modifications indicate that SA can be realizable in dynamic environments like the MANETs. Also, we have shown that the proposed modifications do not affect the quality of the clustering solutions, even though these solutions might not be anymore global optimal. The latter is not important in such environments where the topology is constantly changing and the clustering solutions provided by the modified SA are temporary.

Even though we have demonstrated the effect of the initial solution to the convergence time of SA, there is still a very important issue to be resolved for the completion and realization of the approach. This issue is related to the methods that we have to apply in order to obtain quality initial solutions. Our objective is to present such methods and prove that the combination of these methods with SA still provides better convergence time compared to the SA, which relies on a random initial solution.

REFERENCES

- Baker D., Ephremides A., and Flynn J., "The design and simulation of a mobile radio network with distributed control," IEEE Journal on Selected Areas in Communications, SAC-2(1):226--237, 1984
- Basagni S., "Distributed and Mobility-Adaptive Clustering for Multimedia Support in Multi-Hop Wireless Networks," VTC 1999-Fall, Vol. 2, pp. 889-893
- Chatterjee M., Das S. K., Turgut D., "WCA: A Weighted Clustering Algorithm for Mobile Ad hoc Networks," Journal of Cluster Computing (Special Issue on Mobile Ad hoc Networks), Vol. 5, No. 2, April 2002, pp. 193-204
- Johnson D.S. and McGeoch L.A., "The Traveling Salesman Problem: A Case Study in Local Optimization," in E. H. Aarts and J. K. Lenstra (eds.), "Local Search in Combinatorial Optimization," John Wiley and Sons, Ltd., pp. 215-310, 1997
- Kirkpatrick S., Gelatt C.D.Jr., and Vecchi M.P., "Optimization by Simulated Annealing," Science 220 (13 May 1983), 671-680
- Lin R. and Gerla M., "Adaptive Clustering for Mobile Wireless Networks," IEEE Journal on Selected Areas in Communications, pages 1265-1275, September 1997
- Manousakis K., Baras J.S., "Dynamic Clustering of Self Configured Adhoc Networks Based on Mobility ", Conference on Information Sciences and Systems (CISS) 2004, Princeton, New Jersey, March 17-19, 2004
- Manousakis K., McAuley A. J., Morera R., "Applying Simulated Annealing for Domain Generation in Ad Hoc Networks," International Conference on Communications (ICC 2004), Paris, France, 2004

Manousakis K., McAuley A. J., Morera R., Baras J.S.,
“Rate of Degradation of Centralized Optimization
Solutions and its Application to High Performance
Domain Formation in Ad Hoc Networks,” MILCOM
2004, Monterey, CA, USA, November 2004.

Manousakis K., McAuley A.J., Morera R., Baras J.S.,
“Routing Domain Autoconfiguration for More
Efficient and Rapidly Deployable Mobile Networks,”
Army Science Conference 2002, Orlando, FL

Su W., Lee S.-J., and Gerla M., "Mobility Prediction and
Routing in Ad Hoc Wireless Networks," International
Journal of Network Management, 2000

The views and conclusions contained in this document are those of the
authors and should not be interpreted as representing the official
policies, either expressed or implied of the Army Research Laboratory
or the U.S. Government