RATE OF DEGRADATION OF CENTRALIZED OPTIMIZATION SOLUTIONS AND ITS APPLICATION TO HIGH PERFORMANCE DOMAIN FORMATION IN AD HOC NETWORKS

Kyriakos Manousakis, John S. Baras
Electrical and Computer Engineering
and the Institute for Systems Research
University of Maryland, College Park
College Park, MD, 20740

and

Anthony J. McAuley, Raquel Morera
Telcordia Technologies Inc.
Piscataway, NJ, 08854

ABSTRACT

Future military systems such a FCS and WIN-T require a robust and flexible network that supports thousands of ad hoc nodes; therefore, we must ensure the scalability of networking protocols (e.g., routing, security and QoS). The use of hierarchy is a powerful solution to the scaling problem, since it allows networking protocols to operate on a limited number of nodes, as opposed to the entire network. We have proposed an automated solution to dynamically create and maintain such hierarchy based on a combination of global optimization algorithms [1] and local distributed maintenance protocols [2]. Global optimization clearly improves performance in a static network but, it is unclear how effective it is in a dynamic ad hoc environment. In this paper, we analyze how the hierarchy created deteriorates from the optimal as network conditions change. We show that the fragility of the optimization depends on the particular cost function and the number of metrics that change. More important, we show, for the first time, that global optimization can remain effective for long periods with good cost functions, even in large dynamic ad hoc networks (where metrics may change rapidly due to node mobility and links making and breaking). This result shows that, with fast optimization algorithms such as modified Simulated Annealing [1], future military systems can use global optimization to autoconfigure domains to significantly improve performance. We also show that local maintenance protocols support the global optimization mechanisms by extending the time the hierarchy remains feasible.

I. INTRODUCTION

If heterogeneous ad hoc battlefield networks are to scale to hundreds or thousands of nodes, then some form of hierarchy is needed. One technique is to dynamically create a good hierarchy using Domain Autoconfiguration. Domains allow routing, QoS and other networking protocols to operate on fewer nodes, with cross-domain interaction only through a few border nodes. This division greatly reduces overall overhead (e.g., routing overhead with n nodes goes from $O(n^2)$ to $O(n \log n)$).

To provide this hierarchy many dynamic clustering algorithms, mainly based on local distributed approaches, have been proposed in the literature [5] [6] [7] [8]. Their drawback, however, is that they do not take into consideration the overall network environment. Indeed, in many cases, these algorithms harm network performance instead of improving it because of the reclustering overhead they impose in a dynamic network. We have proposed an automated way to dynamically create a good hierarchy using Domain Autoconfiguration based on a combination of global optimization [1] and local distributed maintenance [2]. This allows selection of domains to ensure a global optimization and allows protocols to be placed in domains tuned to more homogenous conditions [3]. The centralized optimization algorithm relies on a set of cost functions [1] that are selected appropriately based on the network environment and the desired performance.

Using global network information, in addition to the local maintenance, appears counter-intuitive for ad hoc networks. Although the centralized global optimization provides significant benefits (i.e., obtains the most optimal clustering map for the given cost function) when first configured, it must be shown how effective global optimization is in a dynamic ad hoc environment. To minimize overhead, global optimization cannot run continually. Thus, it may use incomplete, stale, or even inaccurate metrics. It is therefore important to analyze how quickly the optimization deteriorates as the variables (network conditions) change. In particular, we must know how the optimality degrades with time in dynamic networks. Also, if there is no local domain maintenance algorithm (e.g., [2]), then we must see how fast the solution becomes infeasible.

Indeed, by the time the information is collected, the optimization process terminates and configuration information is distributed, we found that in some cases the clusters generated by the algorithm are no longer optimal (and possibly infeasible). Thus, though we have made significant progress on improving the optimization time [1], it is critical we understand how quickly the optimality degrades over time in a dynamic network.

This paper will present the first results showing how optimality degrades over time for centralized domain algorithms. We show the dependence on the cost functions selected for the optimization algorithm and the mobility characteristics of the participating nodes (the mobility models that we apply are the Random Waypoint Mobility Model and the Reference Point Group Mobility Model). The importance of the dependence on the cost functions is that if we cluster in a way to produce robust clusters (e.g., mobility characteristics of the nodes) then the optimality degrades slower over time compared to the case where we cluster independently of the mobility characteristics of the nodes (e.g., cluster size).

The next section presents an overview of our centralized domain generation protocol, which consists of the Simulated Annealing algorithm and a set of cost functions and constraints. Section III will describe the importance of determining the convergence time requirements and the factors that affect these bounds. Section IV presents the convergence time characteristics of SA and the method we applied for measuring the convergence time bounds along with their corresponding values. The last section concludes the paper and provides some directions for future work.

II. DOMAIN OPTIMIZATION USING GLOBAL INFORMATION

This section presents our domain optimization approach based on using global information [1] with various cost functions and topological constraints. We use a modified Simulated Annealing algorithm, but describe it only briefly, since the results in this paper are independent of the particular choice of algorithm. However, we will describe in detail ten different cost functions, since the rate of change of optimality depends heavily on these cost functions. We also describe the topological constrains, since, without local domain maintenance (e.g., [2]), the constraints affect the feasibility of the solution.

A. SIMULATED ANNEALING

Simulated annealing (SA) has been widely used for tackling different combinatorial optimization problems [9]. 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. The description of our modified SA algorithm is described in detail in [1], but Figure 1 summarizes its operation.

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*. A key characteristic of simulated annealing is that it allows uphill moves at any time and relies heavily on randomization.

Inputs	Examples
Equilibrium function	Constant (j = 5000); "Stop repeats"
Cost function	$\sum^{x} Diameter(C)$
Cooling function	Geometric or logarithmic
Stop function	Minimum temperature. "Stop repeat" criteria
Reclustering	Random move of one node
T0	Initial Temperature
K	Number of clusters

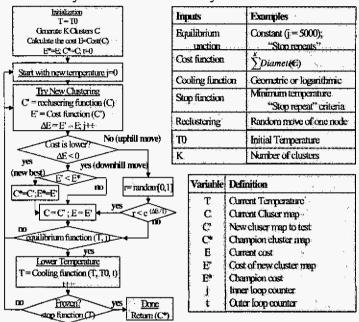| Variable	Definition
T	Current Temperature
C	Current Cluster map
C'	New cluster map to test
C*	Champion cluster map
E	Current cost
E'	Cost of new cluster map
E*	Champion cost
j	Inner loop counter
t	Outer loop counter

Figure 1 Simulated Annealing algorithm for network partitioning

From the point of view of this paper the important result is that the SA produces the optimal (or near optimal) Clustering C^* with the lowest Energy E^*. We will measure how this Energy E^* changes over time as the metric change (i.e., nodes move) without any re-optimization.

B. METRICS

In this section we present the set of metrics that will be used in our cost functions. The metrics can be categorized in two large classes. The first class of metrics is related to the network environment characteristics [1]:

- **Cluster Size** $|C_i|$: The number of nodes that have been assigned to the cluster. Minimizing Cluster Size reduces the overhead and improves the performance of most networking protocols. For example, we know that the overhead of most routing protocols is proportional to the square of the number of nodes.

- **Cluster Diameter** d_{C_i}: The size of the longest path within a cluster in number of hops. Minimizing diame-

ter can reduce overhead and latency of many networking protocols. For example, a smaller diameter allows proactive routing protocols, which exchanges routing information among all nodes, to update information (e.g., link failure) quicker and with less overhead.

- **Border Routers** BR_{C_i} : The number of nodes that interconnect two or more clusters. There are scenarios where we want to have some minimum number of border nodes to improve robustness or provide more bandwidth for inter-domain communication. In other cases we want to minimize the number of border nodes, to minimize inter-cluster signaling.

The second class contains the metrics related to the node's mobility characteristics [10]:

- **Direction** θ_i : The direction of a node described as the angle counter-clockwise from the straight from two consecutive points on the trajectory of the node and the straight line parallel to the positive x-axis (see Figure 2). A node can estimate its direction of movement utilizing various tools, such as a GPS device.
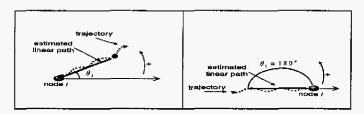

Figure 2 Definition of θ_i

- **Speed** U_i : The speed of a node i is the magnitude of the distance that is covered in a unit of time (meters per second). A node can estimate its speed using a GPS device or by other means.

- **Relative Direction** $\theta_{r_{ij}}$: The relative direction of two nodes. If $node_i$ is moving with direction θ_i and $node_j$ with direction θ_j then:

$$\theta_{r_{ij}} = \min\left(\left|\theta_i - \theta_j\right|, 360 - \left|\theta_i - \theta_j\right|\right), \quad (1)$$

Note that $\theta_i, \theta_j \in \left[0^o, 360^o\right)$, $\theta_{r_{ij}} \in \left[0^o, 180^o\right]$. Figure 3 gives two graphical examples showing the computation of the relative direction $\theta_{r_{ij}}$ of two nodes i and j.
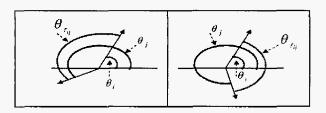

Figure 3 Relative Direction of two nodes i and j

- **Relative Velocity** $U_{r_{ij}}$ The relative velocity of two nodes is the velocity with which a node approaches or recedes from another node:

$$U_{r_{ij}} = \sqrt{(U_i \cos\theta_i - U_j \cos\theta_j)^2 + (U_i \sin\theta_i - U_j \sin\theta_j)^2} \quad (2)$$

- **Link Expiration Time** LET_{ij} : The Link Expiration Time is defined as the estimated lifetime of the link that connects two nodes i and j. Figure 4 shows an example of the calculation of LET_{ij} for two nodes at coordinates (x_i, y_i) and (x_j, y_j).

$a = u_j \cos\theta_j - u_i \cos\theta_i$

$b = x_j - x_i$

$c = u_j \sin\theta_j - u_i \sin\theta_i$

$d = y_j - y_i$

$r = TxRange_{j,i}$ (TxRange in this case is assumed the same for every node)
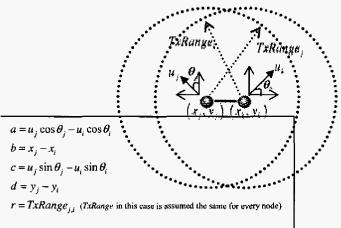
Figure 4 Life Expiration Time for a link between two nodes

For the example given in Figure 4 it can be shown [11] that:

$$LET(j,i) = D_{(j \to i)} = \begin{cases} \dfrac{-(ab+cd) + \sqrt{(a^2+c^2)r^2 - (ad-bc)^2}}{a^2+c^2} & \text{,nodes } j,i \text{ are in range} \\ 0 & \text{,nodes } j,i \text{ are not in range} \\ \infty & \text{,nodes } j,i \text{ are relatively static} \end{cases} \quad (3)$$

C. COST FUNCTIONS

Simulated Annealing is one of many global optimization algorithms that we can utilize to obtain optimal or suboptimal clustering decisions [9]. 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 objectives [1] [10].

Table 1 Example Cost Functions and Network Objectives

Objective	Cost Function
Balanced Size Clusters	$J(C) = \min\left(Var\left(\lvert C_1\rvert^2, ..., \lvert C_K\rvert^2\right)\right)$ (1) $J(C) = \min\sum_{i=1}^{K}\lvert C_i\rvert^2$ (2)
Balanced Diameter Clusters	$J(C) = \min\sum_{i=1}^{K}\left(d_{C_i}^2\right)$ (3) $J(C) = \min\left(Var\left(d_{C_1}^2, d_{C_2}^2, ..., d_{C_K}^2\right)\right)$ (4)
Balanced Size Clusters with the minimum number of Border Routers..	$J(C) = \min\left(Var\left(\lvert C_1\rvert^2, ..., \lvert C_K\rvert^2\right) + \left(\sum_{i=1}^{K}BR_{C_i}\right)\right)$ (5)
Cluster members move in a similar direction, so we expect longer durations of stable cluster membership	$J(C) = \min\left[\sum_{z=1}^{K}\left(\sum_{i,j=1}^{\lvert C_z\rvert}\theta_{r_{i,j}}\right)^2\right]$ (6) $J(C) = \min\left[\sum_{z=1}^{K}Var\left(\theta_{r_{i,2}}, ..., \theta_{r_{\lvert C_z\rvert\cdot\lvert C_z\rvert}}\right)\right]$ (7)
Cluster members have similar velocity, so we expect more stable cluster membership	$J(C) = \min\left(\sum_{z=1}^{K}\left[\sum_{i,j=1}^{\lvert C_z\rvert}U_{r_{i,j}}^2\right]^2\right)$ (8)
Links among cluster members have long expiration time estimates. Improves the lifetime of the generated hierarchy	$J(C) = \min\left[-\sum_{z=1}^{K}\left(\sum_{i,j=1}^{\lvert C_z\rvert}I_z\left(LET_{ij}\right)\right)^2\right]$ (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)	$J(C) = \min\left(\sum_{z=1}^{K}\left[\sum_{i,j=1}^{\lvert C_z\rvert}\frac{U_{r_{i,j}}}{2*\max(S)} + \sum_{i,j=1}^{\lvert C_z\rvert}\frac{\theta_{r_{i,j}}}{180}\right]^2\right)$ (10)

Parameter	Definition
C_i	Cluster i
$\lvert C_i\rvert$	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

D. TOPOLOGICAL CONSTRAINTS

The clustering decisions where the optimization algorithm (e.g., Simulated Annealing) searches for the optimal clustering is limited by the requirements of the domain topology. In particular we want a node within a cluster to be able to reach all other members of the cluster without passing outside the cluster. More formally we define a topological cluster as a set $\mathbb{S}$ of nodes where for $\forall node_i, node_j \in \mathbb{S}$ 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 $node_k \notin P_{ij}$. The constraint of topological clusters is important, since we want the members of the generated clusters to be isolated from the members of other clusters.

III. TIME TO CALCULATE THE OPTIMAL DOMAINS

This section looks at the time required to calculate the optimal domains, using a modified Simulated Annealing algorithm, for different network sizes and numbers of generated domains.

A. RELEVANCE OF RUNNING TIME FOR OPTIMIZATION

The time it takes for the optimization to complete does not affect the rate of degradation of the optimality (or how quickly the solution can become infeasible); however, the speed of optimization does place a lower bound on goodness of centralized optimization. For example, if it takes two minutes to generate the optimal solution and the solution becomes infeasible after one minute we should not consider central optimization. In other words, if the optimization operates on metric values collected t_c seconds before, the optimization itself takes t_o seconds and the distribution of the new configuration takes t_d seconds, then the result has already degraded for $t_c + t_o + t_d$ seconds. As t_c and t_d are typically not under our control, we investigate the optimization time t_o.

As we have shown in [1] the important parameters that determine the convergence time characteristics of Simulated Annealing algorithm is the applied cost function, the number of generated clusters, the selection of cooling schedule and the termination condition of the algorithm (e.g., *StopRepeats* value). Note, however, that even though this section calculates optimization time t_o based on Simulated Annealing, this does not affect the results for degradation of the optimality or how quickly the solution can become infeasible.

B. OPTIMIZATION TIME RESULTS

Figure 5 shows the time it takes for the Simulated Annealing algorithm to run for the first cost function shown in Table 1. The results were obtained on a 700MHz Pentium III processor with 256MB RAM, which was running Linux (kernel v. 2.4.20-6). It shows the results for different number of nodes in the network (from 100 to 1000) and for different numbers of generated clusters (from 2 to 10).

Convergence Time VS Nodes VS Clusters

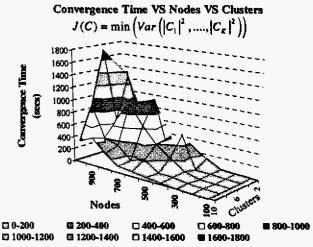$$J(C) = \min\left(Var\left(|C_1|^2, ..., |C_K|^2\right)\right)$$

Figure 5 Convergence Time of SA algorithm with respect to the network size and the number of generated clusters

The general observations are that the larger the network size and the smaller the number of generated clusters the higher the convergence time of SA. The convergence time decreases as the number of generated clusters increases because for a specific network size the more the number of generated clusters the less the potential clustering solutions. By analyzing more these results, we can conclude that the size of generated clusters has the most significant impact on the convergence time. This is because the cluster size parameter depends both on the network size and the number of generated clusters. For smaller cluster sizes the fewer are the potential clustering solutions to be evaluated from the SA algorithm, which results in shorter convergence times.

IV. RATE OF DEGRADATION OF OPTIMALITY

The section presents results on the rate of degradation of optimality with respect to changes in metrics that cause changes in the input to the cost function and topological constraints.

A. PROBLEM

When nodes are mobile, the network topology changes and so do the corresponding metric values. We investigate the rate of degradation of optimality of the cost functions for

given topological constraints due to changes in these metrics. In general, the rate of degradation depends on the:

- **Dynamics of the network to be clustered.** Clearly, the more mobile the nodes and the more independent their movement, the faster the topology changes and the lower the probability the solution is feasible upon the termination of the algorithm.

- **Cost Function.** If we cluster based on the expected mobility characteristics of the nodes, the generated clusters are expected to degrade slower than if expected mobility is ignored (e.g. cluster based only on cluster size).
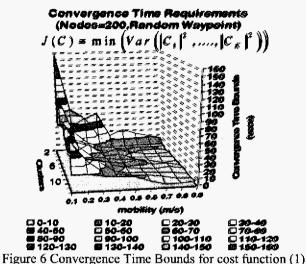
Due to the generality of the method, results here obtain apply to other centralized optimization algorithms. For the characterization of the network environment we applied two different mobility models:

- **Random Waypoint Mobility Model (RWPM).** In RWPM model the nodes select a random destination within the limits of a pre-specified area. Nodes move to these destinations with constant speed, selected at random between 0 and a pre-specified maximum value. When nodes reach their destinations, they immediately select new destinations and new speed.

- **Reference Point Group Mobility Model (RPGM).** In RPGM we define a number of Reference Points (RPs) equal to the number of mobility groups we want to establish. Each node is then assigned to a RP. The movement of the nodes is characterized from the mobility patterns of their corresponding RPs. These mobility patterns are assigned manually to the various RPs in the form of trajectories. When a RP moves to a new location each corresponding node is assigned to a random radius and direction around the new position of the RP. Because of the functionality of RPGM model and the randomness in the selection of the new node position, it is obvious that nodes that belong to the same group may have different speeds and directions.

The input to the method is a random placement of nodes. Next, the optimization algorithm decides the clustering map. We then apply a mobility model to the nodes and recalculate the energy function as the links between nodes make and break.

B. CONVERGENCE TIME BOUND IN THE ABSENCE OF LOCAL DOMAIN MAINTENANCE

Once the clustering decision is made we change the network topology according to one mobility model. The convergence time bound is defined as the time it takes for a clustering decision to become infeasible because the clusters do not satisfy the constraint of constructing

topological clusters. Figures 6 and 7 represent the convergence time requirements for the cost functions (1) and (8) respectively in the case where the nodes are moving in accordance to the Random Waypoint Mobility Model. The maximum allowable speed was varied between 0.1 m/s to 1 m/s and there was no pause time assumed. The number of generated clusters was varied from 2 to 15 clusters and the network size is 200 nodes.

Convergence Time Requirements
(Nodes=200,Random Waypoint)

$$J(C) = \min\left(Var\left(|C_1|^2, ..., |C_K|^2\right)\right)$$

☐ 0-10	☒ 10-20	☐ 20-30	☐ 30-40
☒ 40-50	☐ 50-60	☒ 60-70	☐ 70-80
☒ 80-90	☐ 90-100	☒ 100-110	☐ 110-120
☒ 120-130	☒ 130-140	☐ 140-150	☒ 150-160

Figure 6 Convergence Time Bounds for cost function (1)

Convergence Time Requirements
(Nodes=200,Random Waypoint)

$$J(C) = \min\left(\sum_{z=1}^{K}\left[\sum_{i,j=1}^{|C_z|} U_{r_{i,j}}^2\right]^2\right)$$

☒ 0-20	☒ 20-40	☐ 40-60	☐ 60-80	☒ 80-100
☐ 100-120	☒ 120-140	☐ 140-160	☒ 160-180	☒ 160-200
☐ 200-220	☐ 220-240	☒ 240-260	☒ 260-280	☒ 280-300

Figure 7 Convergence Time Bounds for cost function (8)

The convergence time bound on the clustering algorithm is much stricter for cost function (1) than for cost function (8). The objective of cost function (1) is to generate balanced size clusters, thus it does not take node mobility into consideration, while (8) generates robust (long-lived) clusters by grouping nodes with similar mobility characteristics. We conclude that, to extend the applicability of centralized algorithms to dynamic networks, cost functions must take into account the dynamics of the nodes.

Figure 8 shows the ratio of feasible clustering decisions taken by SA at the time the algorithm terminates as a function of node mobility. A cluster configuration is unfeasible if it violates the topological cluster requirement.

Acceptance Ratio of Clustering Maps
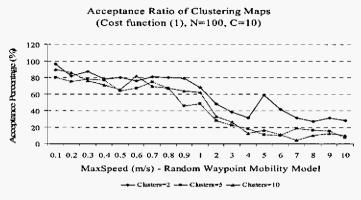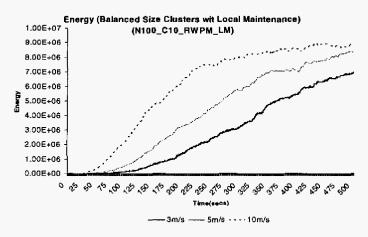**(Cost function (1), N=100, C=10)**

Figure 8 Acceptance (%) of the clustering decisions subject to RWPM model
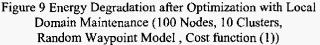
Interestingly we found a case where the acceptance percentage stays constant. In this case we applied the RPGM model where we assumed two mobility groups (50 nodes each) that were moving towards the same direction but with an average relative speed of 4m/s. Due to the different mobility characteristics presented by the nodes of the two mobility groups, the application of cost function (8) results in accurate identification of the two mobility groups. In this case the clustering decisions were always feasible. The nodes of these groups continue to move together through time, so the clustering is always the optimal one with respect to the mobility cost function.

C. CLUSTERING DEGRADATION RATE WITH LOCAL MAINTENANCE

In the previous subsection we assumed no local maintenance. We have proposed an automated creation and maintenance of a hierarchy based on a combination of global optimization algorithms [1] and local distributed maintenance protocols [2]. For example, implementing a simple local maintenance algorithm a node that gets disconnected from its cluster can join another cluster; if the node can join more than one cluster without violating the feasibility criterion it selects the one with the lowest cluster ID. We assume the local maintenance protocol is able to maintain connected clusters; but will not be able to maintain optimality. Therefore, we must investigate how the goodness of the optimization deteriorates.

We propose to use the behavior of the energy (cost) function to measure this degradation. These results can indicate the time intervals at which the optimization must run and, indeed, whether it is worth doing any global optimization.

Figure 9 shows how energy (cost) degrades as the time progresses for cost function (1) (i.e. balanced size clusters) and the RWPM mobility model. There are three curves represented in Figure 9, each representing a different maximum speed 3 m/s, 5 m/s and 10 m/s. As expected it is observed that the optimality degrades fast for higher node mobility.

Figure 9 Energy Degradation after Optimization with Local Domain Maintenance (100 Nodes, 10 Clusters, Random Waypoint Model , Cost function (1))

V. CONCLUSIONS

The paper shows that centralized algorithms can be usefully applied to create better domains for even dynamic ad hoc networks. Even though applying centralized algorithms based on global optimization seems counter intuitive, the observations we make in this work show that there are many scenarios where the algorithm can be used with great success. We show the optimization degrades with time and that without local domain maintenance the optimization can quickly become infeasible. However, we also show that even with a simple local domain maintenance algorithm (e.g., [2]) the clustering does not become infeasible and the degradation is gradual. Moreover, we show that by choosing cost functions that select domains based on mobility [10], the rate of degradation in time can be kept much smaller. With higher mobility we need cost function takes into account the dynamics of the nodes, so clusters optimality degrades is not too fast that it requires frequent optimization with high computational and bandwidth overhead. We believe the results show the domain optimization can be applied with great benefit in future dynamic military networks, such as WIN-T and FCS.

The paper also shows that how the convergence time of Simulated Annealing optimization on a modest 700MHz processor allows hundred of nodes to be clustered. We also show that the SA convergence time is proportional to the network size but counter proportional to the number of generated clusters. Even though we focus on Simulated Annealing, this class of results can be generalized for any clustering algorithm due to the independence of the method we applied to derive them. The results of this work can be used as a reference point for the application of any algorithm based on global information in a time sensitive dynamic ad hoc environments.

REFERENCES

[1] K. Manousakis, A. J. McAuley, R. Morera, "Applying Simulated Annealing for Domain Generation in Ad Hoc Networks," International Conference on Communications (ICC 2004), Paris, France, 2004

[2] R. Morera, A. McAuley, L. Wong, "Robust Router Reconfiguration in Large Dynamic Networks," MILCOM 2003, Boston, MA, USA October 2003.

[3] K. Manousakis, A.J. McAuley, R. Morera, J. Baras, "Routing Domain Autoconfiguration for More Efficient and Rapidly Deployable Mobile Networks," Army Science Conference 2002, Orlando, FL

[4] R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," IEEE Journal on Selected Areas in Communications, pages 1265-1275, September 1997

[5] D. Baker, A. Ephremides, and J. Flynn "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

[6] M. Chatterjee, S. K. Das, D. Turgut, "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

[7] S. Basagni, "Distributed and Mobility-Adaptive Clustering for Multimedia Support in Multi-Hop Wireless Networks," VTC 1999-Fall, Vol. 2, pp. 889-893

[8] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, Optimization by Simulated Annealing, Science 220 (13 May 1983), 671-680

[9] D. S. Johnson and L. A. McGeoch, "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

[10] K.Manousakis, J.S.Baras, "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

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