Adaptive Immunity through Differential Elasticity

Fatma Mili
Computer Science and Engineering
School of Engineering and Computer Science
Rochester, Michigan, US
e-mail: mili@oakland.edu

Nancy Alrajei
Computer Science and Engineering
School of Engineering and Computer Science
Rochester, Michigan, US
nmalraje@oakland.edu

***Abstract*— Malicious attacks are often targeted to affect the most vulnerable or most critical resources of a system. In sensor networks, because of the large amount of inherent redundancy, the most serious threats are the ones attacking critical paths in the network attempting to break them thus disrupting the overall function of the network. In this paper we define a set of graph properties that characterize the level of vulnerability of specific links. We use these properties to define a bio-inspired model of self-organization and adaptive reorganization that impart networks with resilience in the face of a variety of scenarios from simple power depletion to targeted malicious attacks.**

Keywords- fault-tolerance; managing redundancy; k-connectedness; differential k-connectedness; elasticity.

I. INTRODUCTION

Redundancy has always been the key ingredient used to provide fault tolerance [10]. Sensor networks are no exception [1]. In fact, they are by design redundant since they consist of a set of interchangeable, functionally overlapping sensor nodes. Upon deployment, a sensor network is generally highly redundant through the use of more nodes than strictly necessary. Its level of redundancy decreases as nodes fail, get depleted of their power [6], or fall victims of unintentional or malicious attacks. An important issue when deploying a sensor network is thus in determining the initial density required that will ensure that the network would remain alive throughout the lifetime of the application (from weeks to months or longer) [9]. When the major risk is depletion and accidental failure, it is reasonable to expect that uniformly deploying more nodes is a quasi optimal way to increase the lifetime of the network. When the risk is malicious attacks, on the other hand, just adding more nodes becomes a very weak strategy because node failures are not random. A malicious attacker can choose which nodes to attack and can concentrate efforts at breaking the network by targeting specific areas. In this paper we discuss the issue of using redundancy and managing redundancy in a way to maximize fault-tolerance. The question we address is: "Suppose that I am willing to devote h times more nodes than needed. What is the best use of the additional nodes? More specifically, given an area A of interest, given a set M of nodes, (1) What is the optimal organization for these nodes? (2) How can we get the nodes to self-organize in a way that would be a good approximation of this optimal organization? (3) How can we get the nodes to autonomously keep adapting their organization to faulty nodes in the network?"

This paper is organized as follows: In Section II, we discuss regular patterns of redundancy and metrics used to capture levels of fault-tolerance. We conclude that these regular patterns and associated metrics are inadequate when it comes to increasing tolerance against malicious attacks. In Section III, we propose an alternative organization and associated metrics that are also based on redundancy, but the redundancy is not uniform but concentrated around areas of the network that are either more vulnerable or whose failure is more consequential. In Section IV, we propose distributed algorithms run on the individual nodes that will make the network organize more strategically and reorganize to reflect node failures. We summarize and conclude in Section V.

II. TRADITIONAL MEASURES OF REDUNDANCY

We consider a sensor network used to monitor some parameters (e.g., detect human/animal presence) over an area (assume a square) A of interest. We further assume that all nodes are identical with a sensing range radius R_s and a communication range R_c. There are typically two concerns: Coverage and Connectivity. Coverage refers to the fact that every point of the area A is within sensing range (distance no larger than R_s) of some node. Connectivity refers to the fact that between any pair of nodes (n, n') there is a sequence n_0=n, n_1, n_2, … n_k=n' such that the distance between any n_i, n_{i+1} is no longer than the communication range R_c; in other words, any two nodes are able to communicate with each other. The network created by the set of nodes defines a graph whereby the nodes are the vertices of the graph and there is an edge between any two nodes that are within communication range of each other. When coverage is the only concern, the minimal number of nodes required to cover area A is proportional to the ratio L^2/ R_s where L is the length of the sides of the square area. This minimal number of nodes can be obtained by organizing the nodes in a lattice of equilateral triangles of side $\sqrt{3}\ R_s$ as is shown in Figure 1. When we are concerned with both connectivity and coverage, the optimal configuration depends on the relationship between R_s and R_c. If the communication range is equal to or larger than $\sqrt{3}\ R_s$, the optimal configuration for coverage more than satisfies connectivity [4,9]. The lattice, in fact will ensure that every node is at the center of a

 ISBN: 978-1-61208-109-0

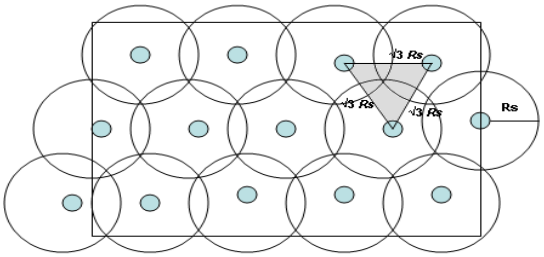

Figure 1. Optimal organization for coverage.

hexagon, and thus is connected to the six corners of that hexagon thus providing a high level of connectivity. In many applications, notably those where the phenomenon being monitored is continuous and not discrete, R_c becomes the most critical parameter and connectivity the most critical issue. This is the subject of our focus here.

To ensure connectivity, when the sensing range is large enough, it is sufficient to organize the nodes in a chain that meanders through the surface. Resilience and fault tolerance require more than simple coverage and connectivity. We need connectivity even after nodes fail, exhaust their power, or get corrupted. In other words, we need to ensure that nodes are connected in more than one way so that when some of these ways fail others remain available. This concept of multiple and redundant connectivity has been identified in the literature under the name of k-connectivity and formally defined below.

The vertex connectivity $K_v(G)$ of a connected graph G is the minimum number of vertices whose removal can either disconnect G or reduce it to a 1-vertex graph.

A graph G is k-vertex-connected if G is connected and Kv(G) >= k. If G has non-adjacent vertices, then G is k-connected if every vertex-cut has at least k vertices.

The edge connectivity $K_e(G)$ of a connected graph G is the minimum number of edges whose removal can disconnect G. A graph G is k-edge-connected if G is connected and $K_e(G)>=k$. G is k-edge-connected if every edge cut has at least k edges. In the case of sensor networks, failures happen at the vertex level rather than at the edge level, thus we are interested in k-vertex-connectivity rather than k-edge-connectivity. In the reminder of this paper we will simply talk about k-connectivity to refer to k-vertex-connectivity. In a k-connected graph, between any two nodes there are at least k disjoint paths [14]. In a k-connected graph, every cut has at least k vertices.

The concept of k-connectivity has been introduced in graph theory and predates the widespread of sensor networks. With the emergence of sensor networks and their applications, many researchers identified the potential of this property for fault tolerance and fault repair [2]. One key challenge in organizing a sensor network so as to obtain the highest connectivity with the minimal number of nodes is that most of these problems are NP hard [5]. For this reason, many researchers in sensor networks, in addition to resorting to simplifying, but application-adequate models [2], focus on finding linear time approximation algorithms. Bredin *et al.* [5] investigated heuristic algorithms for deciding which nodes to wake when some nodes fail and for identifying locations where additional nodes should be placed. Wu and Li [12] and Alzoubi *et al.* [2] focus on the connectivity, not of the whole network, but of the subset of nodes that play a key role in routing. Specifically, they propose approximate algorithms for k-connected m-dominating sets. In [4], Bai *et al.* calculate the optimal deployment for the purpose of achieving k-connectivity. They discuss various patterns such as the triangular lattice and square grids. In [13], Xing *et al.* discuss protocols for ensuring multiple coverage and connectivity. In particular, by using a communication range twice as large as the sensing range, they distribute the nodes so as to ensure coverage, and by the same token have a high level of connectivity. In [8], Khelifa *et al.* discuss the inadequacy of traditional k-connectivity as a true measure of the level of redundancy and level of fault tolerance of a network, notably for large networks. In effect, as the network grows in size, the probability that the k nodes that fail belong to the same cut becomes very small. Conditional connectivity measures the level of fault tolerance in a network by focusing on the probability of failure of all the nodes of the same cut. While Khalifa *et al.* focus on the quantitative aspects, we address here the pragmatic aspects of organizing and reorganizing the network to maximize fault-tolerance. The approach proposed in this paper shares the underlying premise of Ammari and Das [3] by accounting for the conditional probability that the failure of a node will indeed lead to a failure of the network. We motivate our approach by first introducing three intuitive concepts.

Given a square surface area of side length L, and given a set of nodes N, what configuration would statistically maximize the length of time that the network is connected? The rationale here is that given a value k (e.g., 5), it may not be the best strategy to strengthen connectivity in a uniform way as not all vertices are equally useful and not all vertices are equally vulnerable. We discuss multiple considerations when considering the importance and vulnerability of the vertices.

The level of redundancy of a node. In a graph that is 1-connected, there is no redundancy. Every node failure can disconnect the graph. In a k-connected graph, the k-1 first failures are safe because every vertex-cut has at least k vertices. When all cuts have exactly k vertices, and if we call p the probability that any node fails within a time period P, the probability that the network gets disconnected within that same time period from that specific cut is p^k. In practice, different cuts have different sizes. The larger is the size of the cut the lower is the *conditional probability* that the network gets disconnected when a node of that cut fails. Nodes that belong to large cuts have a lower conditional probability of causing the network to fail when they fail. They are therefore "less critical" or "lower impact" (more

disposable) than nodes that belong to smaller cuts. In Figure 2, if we only focus on the two cuts shown, node a is less critical than nodes d and e. Node a belongs to a cut with three nodes, should it fail the connectivity will be ensured through c and e whereas a failure of d makes the network connectivity dependant on only one node: e.

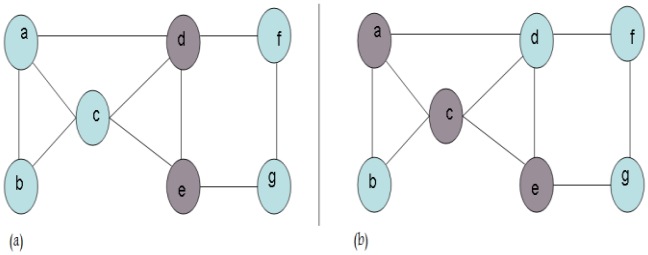

Figure 2. In (a) cut size is 2; in (b) cut size is 3.

Centrality of a node: Chipping vs. chattering the network. Given a connected graph, given a cut that divides the graph in connected components, the number and relative sizes of the resulting connected components are important in determining the impact of such a cut. A cut that barely "chips" the network by isolating a very small connected component from the rest of the network is less damaging than a cut that breaks it apart in many pieces or in two large pieces. The type of breakage that results from a cut characterizes the impact of losing all the nodes of a cut. Nodes in a cut that barely chips are less critical than nodes in cuts that chatter the network. This is illustrated in Figure 3. Losing node d results in disconnecting g but leaving the rest of the network together. By contrast, losing node c breaks the network into 4 components.

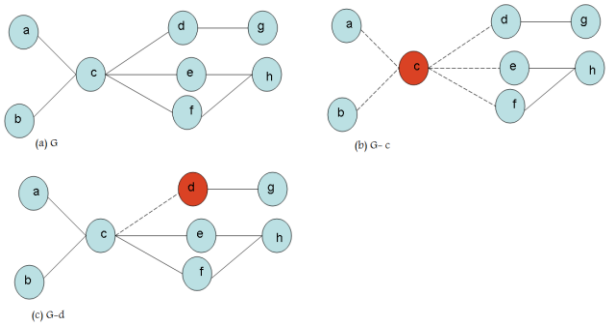

Figure 3. (a) Connected Graph G, (b) G after failure of c, (c) G after failure of d.

"Diameter" of a node: the intensity of the flow that goes through it. The network's functionality is captured by the flow circulating through it. Underlying every non directed sensor network, there is in fact a directed network where most communication takes place from and towards the base station. For example, in the graph in Figure 4, even though all nodes have similar connectivity degrees with communication flowing in both directions, there is an (many) underlying tree showing the routing of information from the base station to the rest of the network and from the rest of the network towards the base station. Thus, what is even more important than the connectedness of a node is the amount of information (flow) that that node is responsible for. For example, the flow of the incoming information (blue routing tree) is shown in the graph where the number associated with every node represents the number of measurements that node transports (effectively, it is the size of the routing sub-tree rooted at that node).

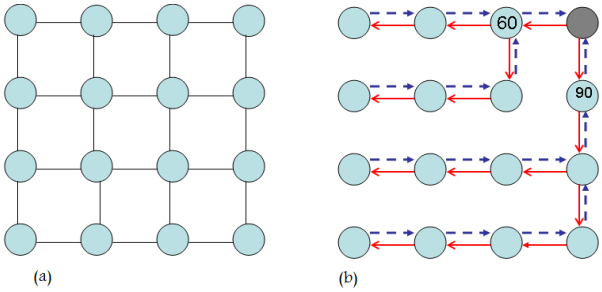

Figure 4. Flow through the network

Before we discuss these 3 concepts further we have to recognize that the first two concepts characterize a node based on "the" cut under consideration. Generally, a node belongs more than one cut, therefore a full characterization of a node would need to combine all cuts. We will not do that.

These concepts are presented here simply as an intuitive motivation for the following section. We are not planning on quantifying them or computing them. What we discuss here instead is the way in which they are inter-related. The level of redundancy of a node is the redundancy that is formally captured by the concept of k-connectivity except that k-connectivity sets an *initial* lower bound and captures the minimal level of redundancy present *in the network*. The conditional probability that the network gets disconnected when a node fails on the other hand, is associated with a node is a local measure of how much back up a node has. The centrality and diameter of a node are related in the sense that the flow that goes through a node is typically the flow from the sub-tree defined by that node towards the base station. The smaller is the sub-tree, the more the failure is likely to result on a chipping; the larger is the sub-tree, the more the failure is likely to result in a chattering. Therefore, both concepts correlate. Measuring one or the other will do. The key is to organize the network so that nodes with a high flow have a high level of redundancy.

III. DIFFERENTIAL REDUNDANCY

A. *Differential Connectivity*

The idea behind differential connectivity is that given a set of nodes available, instead of placing the nodes so as to ensure a uniform size cut everywhere throughout the network, and thus decreasing the conditional probability of

 ISBN: 978-1-61208-109-0

failure of the network if a node fails, we should instead place the nodes so as to decrease the probability proportionally to the size of the flow going through a node. In other words, we want to ensure that nodes with a high volume of flow belong to large cuts. In other words, rather than seeking a uniform level of connectivity, we seek instead a uniform level of

B. *Elasticity*

Regardless of how efficient is the original organization of the network, such organization is only statistically quasi-optimal. If there are no malicious attacks and nodes only fail because they exhaust their power it is possible that the network will remain connected until it collectively dies out. In other words, the network will deplete uniformly and die graciously chipping one bit at a time. Such scenario, while possible, is not very likely. There are enough unknowns to almost guarantee that whatever organization is selected in the beginning, it does not match exactly the scenario that will take place. For this, we want the organization to be dynamic and adaptive by allowing nodes from robust areas to cover for areas that get depleted. This is the concept of elasticity: whenever an area of the network is under excessive pressure, whenever possible, it should stretch rather break. In fact, elasticity consists of stretching much earlier than the time when the network is at the point of breaking. For example, consider the network in Figure 5 below. The flow goes from x to y, then z. Initially x, y, and z have cuts of size 3, 4, and 5 respectively. As 2 nodes from

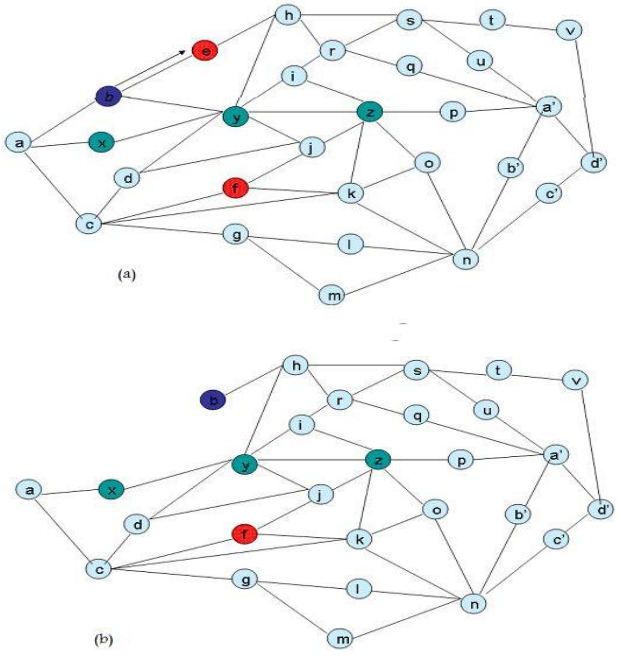

Figure 5. (a) nodes e and f fail they are part of cut set of node y. (b) node b moves to be part of node y's cut set

the cut of y fail, one node from the cut of x moves up and one node from the cut of z moves down resulting in cut sizes of 2, 4, and 5 (the nodes that migrates from z down to y is part of the cut of both y and z).

C. *Differential Elasticity*

The concept of elasticity alone refers to the fact that when some pressure is exercised pulling on a surface from opposite directions, the components of the surface are rearranged as a response in order to avoid breaking the bonds between them and tearing apart. Some surfaces are elastic in all directions; others have a preferred (or exclusive) direction of elasticity. In this case, we want the elasticity to be in the direction of the flow. In other words the "stretching" of the network must be done primarily in parallel with the direction of the flow rather than transversally to it. This direction of movement of the nodes will be in line with where the highest need is likely to be and will also minimize un-necessary back and forth of the nodes as the network thins out.

IV. ADAPTIVE ORGANIZATION

A. *Autonomous Organization*

We recall that the objective is to have the nodes self organize so as to achieve a differential connectivity by giving a larger cut size to nodes with higher flow volume. We recall also that the intention is to use k*N nodes where only N would be sufficient for coverage and connectivity. We propose here a two-step process:

1. A sufficient number of nodes (e.g. 1.5*N) are deployed by spreading them using a uniform random distribution to ensure full coverage and connectivity.
2. This initial set of nodes collaborates to establish a reasonable flow pattern and identify the amount of flow traversing each of the nodes.
3. The remaining nodes are deployed in such a way that areas with high flow will attract a relatively large number of nodes, resulting in larger cuts whereas areas of small flow would attract fewer nodes resulting in smaller cuts.

We discuss each of these three stages:

1. Initial Uniform node deployment. This is the usual method to deploy most large sensor networks. The nodes are sprayed from a distance. A sufficient number of nodes are sprayed to maximize the likelihood that the network will cover the area of interest and will be connected. Because of the vulnerability of nodes (limited-life time battery, among others), most networks are deployed with sufficient redundancy. In other words, when only N nodes are sufficient to cover the area when placed in an optimal configuration, 2 (or more)*N nodes are placed. In this case, we target to provide enough redundancy to ensure connectivity and coverage, but we start with a number close enough to the minimal.

2. Establishing the flow pattern. A number of algorithms and approaches have been proposed to guide the pattern of communication (routing) between the nodes of a network. It is without loss of generality that we will assume that: 1. The

overall routing takes place using a routing tree, sub-graph of the overall network, 2. The tree is rooted at the base station or some other representative sink node, 3. The communication can be decomposed into two flows: from the root of the tree towards the leaf nodes to broadcast the "query", i.e., the nature of the data requested from the nodes and from the leaf nodes up towards the root to send the data back to the base station. 4. The two flows follow the same links in opposite directions. 5. Finally, we assume that a quasi-minimum spanning tree rooted at the sink node minimizing the total communication cost is a good approximation of the actual routing structure that will be used. The spanning tree will be constructed; the flow traversing each of the nodes based on this minimum spanning tree will then be used to estimate the actual flow going through those same nodes in the sensor network. The spanning tree has the advantage of being easily computed using a greedy distributed algorithm.

3. Deploying the remaining nodes. The key now is to deploy the remaining nodes so that they assemble around existing nodes in a way proportional to the flow through them. The general idea is to disperse the nodes and then subject them to attractive forces by the existing nodes whereby the intensity of the attractive forces is correlated with the flow in the nodes. One way to visualize this process is to see the nodes with their flow as points in 3D space where their positions on the plane represent their x and y coordinates and their flow represent their elevation (the elevation is in fact inversely proportional to the flow). Figure 6 shows the topography generated by a network flowing towards the bottom of the area.

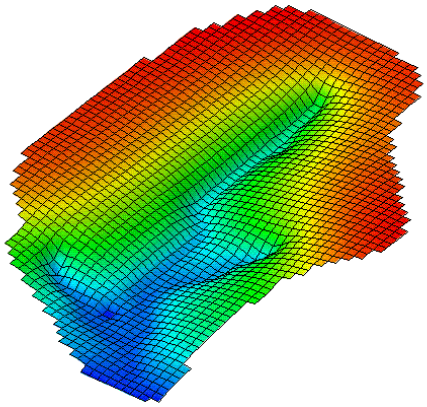

Fig. 6. Topography created by the network

In other words, whereas the first batch of nodes establishes the topography, the remaining nodes are dispersed and let initial random positions, gravity forces, and inertia determine their final positions. We discuss each of these three elements in turn:

Initial positions: In the same way that the initial batch was deployed at random to cover the area, we can do the same thing with the remaining nodes.

Gravity: The forces used to organize the node are calculated based on local information about gravity forces. The mobile nodes in this new batch have a node id, an x- and y- position, but have no flow (yet), i.e., they are not aware of their elevation. This latter information is obtained from the nodes in their communication range. Each node from the first batch broadcasts a message m=<topic= "position", id=self.id, x=self.x, y=self.y, z=self.f>. Each node from the new batch listens to all of the incoming messages, identifies nodes with the "lowest elevation", and moves towards them. The relative positions of the neighbors and their elevations determine the direction of movement and its intensity. More specifically:

- If there is only one neighbor, move towards that neighbor (this is unlikely to happen given the level of connectivity) only if its elevation is lower than self.
- If there are multiple neighbors with lower elevation than self, identify two neighbors with the lowest elevation, move towards the middle between them, calculate own elevation to be higher than the highest of the two.
- If there are multiple neighbors but all of them have a higher elevation then self, do not move.

Each of the movements prescribed above is a small step after which the node listens again and may keep moving.

Inertia: Using the gravity alone, nodes stop only when they reach local minima. Because the flow grows rather uniformly from the boundaries of the area towards the sink node, there is a risk that all the nodes will end up flocking towards the lowest elevation point, i.e., the sink node. We add an inertia force to ensure a more distributed spreading of the nodes. We use a repulsion force based on the crowdedness of the neighborhood. A node attracts mobile nodes when it has a low elevation. Also, a node also repulses mobile nodes with intensity proportional to the crowdedness of its neighborhood.

B. Adaptive Reorganization

Assuming that we start with a reasonably good organization of the network, this organization may no longer be good after repeated failures of nodes. For sensor networks, node failures can be completely random with no correlation between the failure of a node and its position or the failure of its neighbors, or they may be relatively predictable (power depletion of a node with high level of activity), or correlated with the failure of other nodes (an animal tramping on a node may tramp on a cluster of nodes in the same neighborhood, or a malicious attacker who targets specific regions for the specific purpose of disrupting the network). Different patterns of failures require different approaches. Our proposal consists of three complementary approaches.

Differential connectivity initial organization. The whole idea of crowding the areas of intensive activity prolongs the life of the network by delaying its natural death (disruption)

 ISBN: 978-1-61208-109-0

by exhaustion. This also protects areas that are more vulnerable to targeted attacks.

Differential Elasticity. Irrespective of how good the initial organization, eventually some regions of the network will weaken and eventually "break". Differential elasticity is the process by which mobile nodes keep a watch and move to where they are needed. Specifically, nodes are aware of the cuts to which they belong, and strategically pull other nodes towards them as their cut size is reduced. For example, consider Figure 5 again. Initially node x has a cut size 3 with some flow f_x and node y has a cut of size 3 with a flow f_y. With the assumption that the nodes were distributed to ensure differential connectivity, the flow f_x and f_y must be comparable. When node y loses two nodes, e and f, from its cut, this generates an imbalance. Node y reacts by sending a jolt pulling nodes towards it to reinforce its cut. The force generated is characterized by the following features:

- Its direction is vector (x,y).
- Its intensity is proportional to the expression
- (size of cut (x))/(size of cut(y)) * f_y/f_x -1

When the nodes are balanced, the force is zero. In the case of Figure 5, assuming f_y/f_x =1, the force will have an intensity 3/1-1=2.

Notice that this force will be broadcast to the whole neighborhood and is likely to move multiple nodes closer to y. Some of these nodes may become part of the cuts of both x and y. The parameters dictating the exact intensity and duration of these jolts are determined experimentally so that they generate the right level of "elasticity". We want the structure to be sufficiently elastic to allow the necessary restructuring without being too over-reactive.

Global Restructuring through simulated annealing. In addition to the small local reorganizations, a less frequent more global re-organization is used to allow for better redistribution. The global reorganization is a form of simulated annealing whereby

1. Every mobile node is moved at random within a given distance (this is the heating part of simulated annealing).
2. A few iterations of the organization algorithm are then run to let the nodes cool down and settle back.

C. Status and Future Work

While all the components of our proposed approach are in place, we are still in the process of validating it through simulation. In this simulation we set the length of the area, the communication and sensing range, we calculate the minimal number M of nodes needed for connectivity and coverage, and the multiple k that we would like to use. Given N=k*M, we start by placing a number slightly larger than M using a random placement. We then run the distributed greedy spanning algorithm to determine the flow from the nodes towards the sink node. The rest of the nodes are then placed randomly and subjected to the gravity and inertia forces. We have completed this stage of the simulation after iterative refinements and balancing between the gravity and the inertia forces. We now obtain a node distribution with a relatively uniform ratio cut over flow. Our current efforts are focused on the reorganization step especially that we are considering different types of scenarios for which we want the network to be resilient. These scenarios are (a) Scenarios of power consumption. One certain source of node failure is the depletion of its power. For this, we are experimenting with different types of queries: Queries that involve all regions uniformly; queries that are highly targeted requesting data from the same nodes repeatedly, and mixes of queries of the two types. We found that one of the characteristics of power depletion is that it happens gracefully—irrespective of the scenario. This is because power depletion is a continuous phenomenon that gives nodes time to the affected nodes to notify its neighbors that they may need to take action. (b) Scenarios of hardware failure. We are simulating hardware failure by picking nodes at random and declaring them faulty. As long as the death of the faulty nodes happens in areas that are sufficiently crowded, the network reacts as expected and makes up for the dead node. (c) Scenarios of non malicious accidents. We distinguish this scenario from the previous one by the fact that it is more localized in time and space. We are still experimenting with this scenario especially that we want to validate it with realistic data. Initial results we have obtained from the initial experimentations seem to confirm the well-foundedness of our approach. In particular, the networks organize in the way that we expect them to. For the scenarios that we have completed, the networks also reorganize as intended.

Additional refinements and work under way include:

(1) Refinement of the parameters used for the elasticity force and for the simulated annealing so as to optimize the triggers used for reorganization. We want to make sure that the network reorganizes soon enough to avoid a loss of functionality, but also that it does not re-organize prematurely and then undo work done later.

(2) Assessment of the extent to which the initial organization and the reorganization bring about any gains as compared to a fully random organization or some other organization leading to a uniform k-connectivity.

(3) Assessment of the reorganization. In particular, we will compute the distribution of the ratio size(cut)/flow for every node over time. Ideally we want to see that the variance remains relatively stable over time.

V. CONCLUSION

Redundancy has always been the key ingredient used to provide fault tolerance. Sensor networks are no exception. Because in most applications sensor nodes have limited power and are hard to access, they are understood to be relatively vulnerable. The strength of sensor networks comes not from any single node, but from their number and from the way in which they communicate and collaborate.

 ISBN: 978-1-61208-109-0

Redundancy is an integral part of sensor networks. The general idea being that if we put enough nodes, the network should be resilient since whatever nodes die there will be others in their neighborhood. The concepts of k-connectivity have been used in the recent years to quantify this redundancy and fault tolerance. The concept of k-connectivity for example states that in a k-connected network, the network will remain connected after the failure of *any* k-1 nodes. While very useful, this concept does not fully capture how fault tolerant a network is. In particular, we want to know "how likely is a k-connected network to be disconnected after the failure of k, or k+1, or 2k nodes?" Another important and more pragmatic question is the one we address here: Suppose that I am willing to devote h times more nodes than needed. What is the best use of these additional nodes? Should we organize them so that the network has the highest k-connectivity level? We argue here that instead of uniform connectivity, we would be better off increasing the connectivity where it matters the most rather than uniformly. Furthermore, we propose distributed algorithms that allow the nodes to organize themselves in an autonomous manner and to reorganize themselves as dictated by the changing needs of the network. The proposed algorithms are based on forces exercised by nodes over other nodes in their communication range. The interplay between the different forces generates the desired collective behavior. Initial results from the simulation validate our approach in the sense that we are able to generate the desired behavior repeatedly under different scenarios. We are in the process of develop metrics that can capture the behavior in a more objective manner and generating scenarios that would allow us to compute the gain in energy and lifetime as compared with other approaches that use different initial organizations and criteria and different (or no) re-organizations.

REFERENCES

[1] Al-Karaki, J.N., Kamal, A. E. 2004. Routing Techniques in Wireless Sensor Networks: A Survey. *IEEE Wireless Communications*, Dec. 2004, Vol 11, No. 6, pp.6-28, Dec. 2004.

[2] Alzoubi, K. M., Wan, P., and Frieder, O. 2002. Message-optimal connected dominating sets in mobile ad hoc networks. In*Proceedings of the 3rd ACM international Symposium on Mobile Ad Hoc Networking &Amp; Computing* (Lausanne, Switzerland, June 09 - 11, 2002). MobiHoc '02. ACM, New York, NY, pp. 157-164. DOI= http://doi.acm.org/10.1145/513800.513820.

[3] Ammari, H. M. and Das, S. K. 2009. Fault tolerance measures for large-scale wireless sensor networks. *ACM Transaction on Autonnmous and Adaptive Systems* 4, 1 (Jan. 2009), pp. 1-28 doi= http://doi.acm.org/10.1145/1462187.1462189.

[4] Bai, X., Kumar, S., Xuan, D., Yun, Z., and Lai, T. H. 2006. Deploying wireless sensors to achieve both coverage and connectivity. In *Proceedings of the 7th ACM international Symposium on Mobile Ad Hoc Networking and Computing*(Florence, Italy, May 22 - 25, 2006). MobiHoc '06. ACM, New York, NY, 131-142. DOI= http://doi.acm.org/10.1145/1132905.1132921.

[5] Bredin, J. L., Demaine, E. D., Hajiaghayi, M., and Rus, D. 2005. Deploying sensor networks with guaranteed capacity and fault tolerance. In *Proceedings of the 6th ACM international Symposium on Mobile Ad Hoc Networking and Computing* (Urbana-Champaign, IL, USA, May 25 - 27, 2005). MobiHoc '05. ACM, New York, NY, pp. 309-319. doi= http://doi.acm.org/10.1145/1062689.1062729.

[6] DelaRose, J. Y. Liu, L. Mili, A. Phadkee, L. Davilva, 2005. Catastrophic failures in power systems: Causes, Analyses, and Countermeasures. Invited paper. *Proceedings of the IEEE*. Vol. 93, No. 5, pp. 956-964.

[7] Jia, X, Kim, D., Makki, S. Wan, P-J, and Yi, Ch-W. 2009. Power Assignment for k-connectivity in Wireless Ad Hoc Networks. *Journal of Combinatorial Optimization*, Springer, Netherlands, Vol 9, No 2, pp. 213-222. Doi=http://dx.doi.org/10.1007/s10878-005-6858-2.

[8] Khelifa, B. Haffaf, H. Madjid, M. and D. Llewellyn-Jones 2009. Monitoring Connectivity in Wireless Sensor Networks. *International Journal of Future Generation Communication and Networking* Vol. 2, No. 2, June, 2009

[9] Lambrou, Th., Panayiotou, Ch., Felici, S., Beferull, B. 2010 Exploiting Mobility for Efficient Coverage in Sparse Wireless Sensor Networks. Wireless Personal Communications, Vol 54, pp. 187-201, doi:10.1007/s11277-009-9717-0.

[10] Mili, A. Self-Stabilizing Programs: The Fault-Tolerant Capability of Self-Checking Programs, IEEE Transactions on Computers, pp. 685-689, July, 1982.

[11] Schmid, S. and Wattenhofer, R. 2006. Algorithmic models for sensor networks, *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International* , pp. 25-29 April 2006 doi: 10.1109/IPDPS.2006.1639417.

[12] Wu, Y. and Li, Y. 2008. Construction algorithms for k-connected m-dominating sets in wireless sensor networks. In *Proceedings of the 9th ACM international Symposium on Mobile Ad Hoc Networking and Computing* (Hong Kong, Hong Kong, China, May 26 - 30, 2008). MobiHoc '08. ACM, New York, NY, pp. 83-90. doi= http://doi.acm.org/10.1145/1374618.1374631.

[13] Xing, G., Wang, X., Zhang, Y., Lu, Ch., Pless, R., and Ch. Gill. 2005. Integrated coverage and connectivity configuration for energy conservation in sensor networks, *ACM Transactions on Sensor Networks (TOSN)*, v.1 n.1, p.36-72, August 2005.

[14] Zhang, H. and Hou, J. 2004. Maintaining Sensing Coverage and Connectivity in Large Sensor Networks. In NSF International Workshop on *Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wirelsss, and Peer-to-Peer Networks*, 2004.

 ISBN: 978-1-61208-109-0