Proceedings of SPIE Conference, 6-8 August, Vol. 5205, 2003

Emergent Routing Protocols for Wireless Ad Hoc Sensor Networks

Richard Brooks[a], Matthew Pirretti[a], Mengxia Zhu[b] S.S.Iyengar[b]
Pennsylvania State Applied Research Lab[a];
Department of Computer Science Louisiana State University[b]

ABSTRACT

This paper presents distributed adaptation techniques for use in wireless sensor networks. As an example application, we consider data routing by a sensor network in an urban terrain. The adaptation methods are based on ideas from physics, biology, and chemistry. All approaches are emergent behaviors in that: *(i)* perform global adaptation using only locally available information, *(ii)* have strong stochastic components, and *(iii)* use both positive and negative feedback to steer themselves. We analyze the approaches' ability to adapt, robustness to internal errors, and power consumption. Comparisons to standard wireless communications techniques are given.

Keywords: Sensor networks, adaptation, emergence

INTRODUCTION

In this paper we present four distributed adaptation methods for use by wireless sensor networks (WSNs) in Military Operations in Urban Terrain (MOUT) scenarios. Urban scenarios are challenging since obstructions to radio communications may cause the shortest path between two points to not be a straight line. The chaotic nature of MOUT missions means paths are not reliable and are highly variable. Only transient paths may exist. In spite of this, timely communications are required. Our approaches use local decisions to adapt to a constantly changing substrate. The insights we gained by testing these adaptation methods are being used to design and implement wireless routing protocols. The four methods we analyze are: *(i)* Spin Glass, *(ii)* Multi-fractal, *(iii)* Coulombic, and *(iv)* Pheromone.

1.1. *Wireless Sensor Network (WSN) Definition*

A WSN is a set of sensor nodes monitoring their environment and providing users with timely data describing the region under observation. Nodes have wireless communications. Centralized control of this type of network is undesirable and unrealistic due to reliability, survivability, and bandwidth considerations [1]. Distributed control also has other advantages: *(i)* increased stability by avoiding single points of failure, *(ii)* simple node behaviors replace complicated global behaviors, and *(iii)* enhanced responsiveness to changing conditions since nodes react immediately to topological changes instead of waiting on central command. Our definition of a WSN does not preclude nodes being mobile.

1.2. *WSN Applications*

Consider the WSN application in [2], a surveillance network tracks multiple vehicles using a network of acoustic, seismic and infrared sensors. For a MOUT application, it is essential that the user community have timely track information. Figure 1 shows an idealized MOUT terrain. Black squares are walls or buildings that block radio signals. White squares (known as free cells) are open regions allowing signal transmission. Green (yellow) squares are choke points for signals that are open (closed). Finally, red squares are intermittent disturbances that occur at random throughout the sensor field. Random factors are inserted to emulate common disruptions for this genre of network. Each square capable of transmission contains a sensor node. This amounts to having a sensor field with a uniform density. This provides an abstract example scenario approximating situations likely to exist in a real MOUT situation. This allows us to examine multiple adaptation techniques without being distracted by implementation details. After evaluating adaptation at this abstract level, the insights gained can then be used to create more robust routing protocols.

Figure 1. Idealized urban terrain. Blue oval is the data sink.

1.3. C) WSN Requirements

WSN applications will typically require a large number of nodes to adequately determine the number, position, and trajectories of the objects under observation. To be affordable, individual nodes will be inexpensive and thus unreliable. Power consumption is an important issue. Our goal is to design a WSN that is fault tolerant, consumes minimal resources, supports secure message passing, and adapts well to environmental changes.

II. SPIN GLASS

Ising model is the one of the most important models of statistical physics. Its various generalizations have been used to model and analyze many natural phenomena, ranging from biology to social science. Suppose that we have N magnetic spins, $s_i, i = 1,2,3...N$ on a two-dimensional lattice with each pointing to up $(s_i = +1)$ or down $(s_i = -1)$. Each spin interacts with its neighbors and an overall external magnetic field. The quantum Hamiltonian energy for a configuration of spins $\{s_i\}$ is given by:

$$E[\{s_i\}] = -H\sum_{1}^{N} s_i - K\sum s_i s_j \qquad (1)$$

The first term in equation 1) represents the coupling energy between each spin and external field, H is the coupling constant with external field; the second term explains the interaction energy between all neighboring spins, K gives the strength of spin-spin interaction, K is positive for ferromagnetic bond and negative for anti-ferromagnetic bond. If K is positive, spins with parallel directions have lower energy. Spins with anti-parallel directions have lower energy for negative K.

Spin glass, as a particular variation of Ising model, contains both ferromagnetic and anti-ferromagnetic bonds. In most cases, spins point to random directions and no macroscopic magnetic field is formed due to cancellation among these little magnets. However, in some metal such as iron, a lot of little magnet vectors would align up, producing a permanent perceptible macroscopic magnetic field. The phase transition between magnetization and non-magnetization is tuned by kinetic factor, temperature. Energy minimization force dominates under low temperature and a macroscopic magnetization

forms. Entropy maximization force dominates under high temperature and magnetization was washed away. The probability for each possible microscopic spin configuration is defined by Boltzmann distribution function:

$$P[\{s_i\}] = e^{-E(\{s_i\})/KT} / \sum_A e^{-E(A)/KT} \quad (2)$$

$E(\{s_i\})$: Energy of system in state $\{s_i\}$.

$E(A)$: Summation of energy over all possible states of the system.

K : Boltzmann Constant.

T : Temperature in Kelvin degree.

A simplified quantum mechanics Spin glass model, which still retains basic essence of the Ising model but simple enough to be solved and simulated by computer was derived from the real world of iron magnetism to get some insight into the routing problem of WSN. In contrast to normal up and down spin directions, our sensor node viewed as little spins in a two-dimensional urban warfare scenario can point to eight different directions. It is different from classical physics in which a little magnet can point to any directions it pleases. The non-zero energy difference associated with all directions is critical for the large-scale system correlation [8].

We want to establish dynamic data routes from all sensor nodes to the data sink. Due to the radio transmission range limit, data will be relayed to the data sink in a multi-hop manner. First, a dynamic potential field defining the minimum transmission energy to reach data sink is established through local interactions. Then, potential field together with kinetic factor defines spin direction of each cell by following Boltzmann distribution function. Nevertheless, the formidable number of all possible configurations required in the denominator of the function bans us from strictly following Boltzmann distribution function. Simple brute force method would bring up to 8^N possible configurations for a grid with N cells. Instead, by using local information, only eight local configurations are calculated in the denominator in our model. Such simplification not only dramatically reduces computation load, but also excludes global information. For example, we use $T[n_i]$ to represent potential energy value of node n_i. Node n_j, as one of eight neighbors of node n_i, has potential energy value denoted as $T[n_j]$. Probability for node n_i to point to neighbor n_j, $P[n_{ij}]$ is given by Equation (3):

$$P[n_{ij}] = e^{-E(n_{ij})/KT} / \sum_{\substack{k:\,neighbor\\ of\ node\ i}} e^{-E(n_{ik})/KT} \quad (3)$$

A

$E(n_{ij})$: Energy gap if node n_i points to neighbor n_j as $T[n_j] - T[n_i]$.

$E(n_{ik})$: Energy gap for node n_i to point to all its eight neighbors.

All eight probabilities add up to one. In our computer simulation, a random number is generated to see which direction portion it resides in. Then, that direction is picked as the current spin direction. We repeat this decision process for each sensor node in the lattice at each discrete time step. If we sweep the lattice for a sufficiently large number of times, the fraction of times for sensor nodes pointing to a specific direction will be closed to the calculated true probability.

In order to investigate how kinetic factor tunes the overall macroscopic phase in Ising model. We look at the relative probability of state A and state B:

$$P[A]/P[B] = e^{-D/KT} \quad (4)$$

If KT is much larger than the energy gap ($D = E(A) - E(B)$) between state A and state B, the probability of taking either spin direction would be approximately the same and system is in high entropy state. If the KT is much lower than D. The sensor node is far more likely to be in the lower energy state. Generally speaking, low temperature systems favor a better performance routing in terms of hop distance. T is important, because the shortest path is not the only important criteria. A large T may reduce the power drain on choke points by taking longer routes. An extremely low T can protect the system by

reducing oscillations in the system. Moreover, T can be specified on a per-region basis, allowing flexible control over the terrain.

A sequence of "snapshot" of the system will be captured and displayed as a time dependent process in our simulation animation. Our approach works differently from the standard technique for statistical physics simulation, Metropolis algorithm, however, the underlying idea is similar.

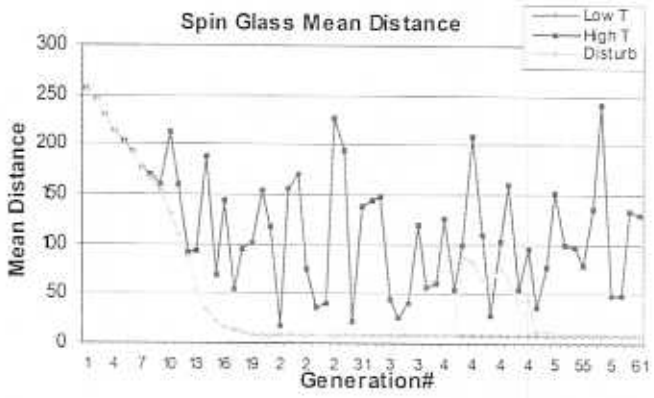

Fig 2 Spin Glass mean distance convergence

To quantify system adaptation we measure the mean distance from each node to the data sink. Fig 2 shows the mean number of hops versus generation number (time step) for a low temperature system (Low T), high temperature system (High T), and a system with a topological disturbance (Disturb). Topological disturbances correspond to choke points in figure 1 opening or closing. The system converges well when T is small, but not when T is large. Topological disturbances are accommodated after a number of fluctuating generations. Fig 3 Further illustrates how mean distances are affected under various temperatures. We observe that there is an abrupt rise in mean hops if temperature is raised above 500 Kelvin.

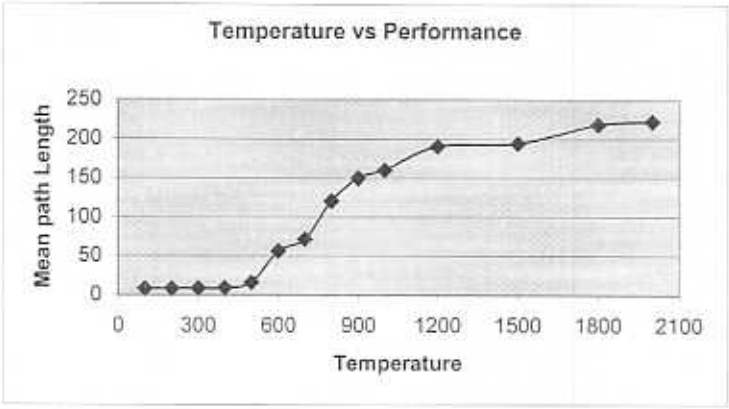

Fig3 Effect of temperature on performance

The amount of message sent during routes establishment phase is quantified to evaluate the scalability of our routing model. We also study how Spin glass model behaves under error conditions. Fig 4 demonstrates how system performance in terms of mean hops is affected by error condition. Error conditions can either be that nodes randomly choose a spin direction instead of following Boltzmann distribution function or nodes send incorrect potential values neighbors. We notice that system is very sensitive to error condition. Performance drastically deteriorates as error begins to occur even at very low level of 1%. Fig 5 illustrates the communication cost versus error conditions. As expected, the amount of messages exchange increases largely due to error message diffusion throughout the system. We conclude that although Spin glass model achieves high performance, the power consumption cost is high and error tolerance is very limited.

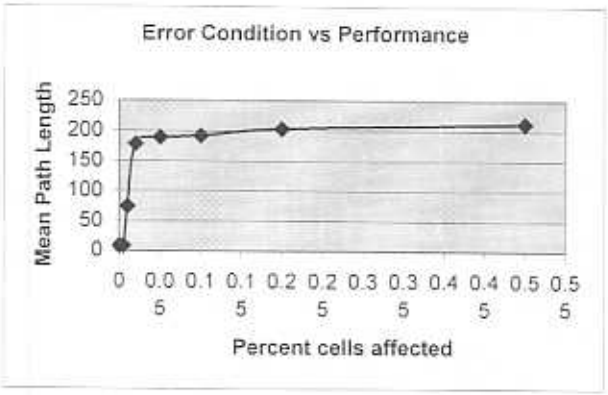

Fig 4 Effect of error condition on performance

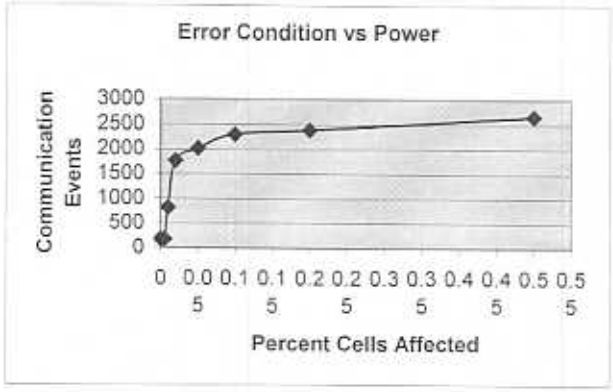

Fig 5 Effect of error condition on power

III. MULTI-FRACTAL

Benoit Mandelbrot father of fractal(1970's) first proposed the notion of fractal, which stands for an irregular geometric object with an infinite nesting of structure at all scales. Examples of fractal objects in nature include coastline, cloud, river and tree. Study shows that multiplicative iteration of random processes creates multi-fractal structure, while additive processes generate mono-fractals [9]. Fractals are objects are characterized by "fractional dimension" and "self-similarity". A single exponent can define mono-fractal, but Multi-fractal can only be depicted by a hierarchy of exponents. People usually call the non-integer exponent index to be fractal dimension.

The classic irreversible fractal growth model for gas and fluid is called Diffusion Limited Aggregation (DLA), first introduced by Witten and Sander(WS) in the early 1980s. Beginning with one foreign seed or even a line segment, random walk of gas or fluid particles become immobilized upon contact with the seeds, if certain crystallization condition is satisfied. Randomly diffusing particles keep sticking to each other and form an aggregate. The structure of this fractal is affected by many factors including crystallization growth inhibition exerted by the crystallization site to the nearby particles. Interfacial surface tension and latent heat diffusion effects can physically explain this inhibition [10]. Such WS like cluster examples can be found in metal electro-deposition experiments. Fig 6 illustrates a crystal structure formed from a straight-line segment.

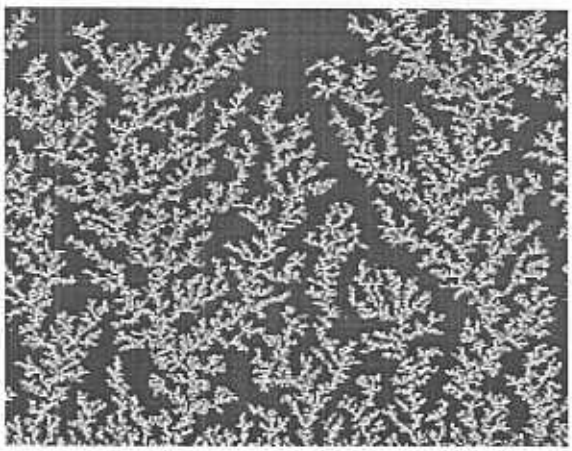

Fig 6 DLA

In our Multi-fractal routing model, data sink is set to be the single foreign seed. Each sensor node vibrates in the lattice without randomly wandering around like gas or fluid particles do, however, we don't exclude the mobility of sensor nodes. A routing tree starts growing from the seed. A sensor node can possibly attach itself to the tree only if any tree nodes reach its neighborhood. Based on the number of neighboring immobilized tree nodes, a set of probabilities of joining the routing

tree is specified. Theoretically speaking, cells are less likely to join in the routing tree as the number of neighboring tree nodes increases. Depending on different level of repulsion effect embedded in the crystallization inhibition parameters, namely the probabilities sets, the growth rate and the routing tree structure can be controlled. Generally speaking, a sparse tree with high region coverage grown in reasonable amount of time steps is desired. In order to select a probabilities set for a good routing tee, a fitness function is constructed to evaluate the quality of the routing trees. For tree i grown under a certain probabilities set, fitness value is computed as:

$$F_i = \frac{C_i}{T_i / b + N_i} \tag{5}$$

C_i : region coverage in percentage

T_i : Discrete time steps.

N_i : number of tree nodes.

b: constant used to normalize time steps and number of tree nodes. The higher the fitness value is, the better the routing tree is. Constant b actually represents our tradeoff between sparsity and routing time.

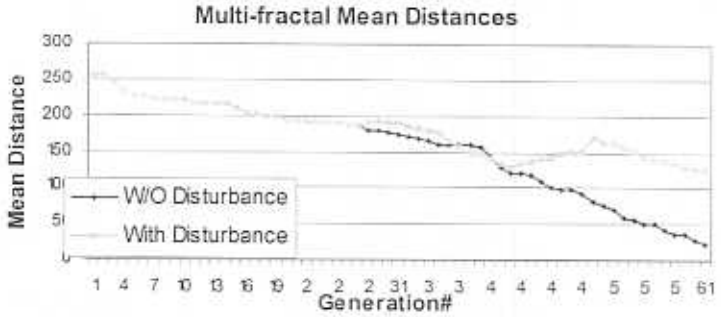

Fig7 Multi-fractal mean distance convergence

Fig7 shows the mean number of hops per generation number (time step) with and without topological disturbances (as for the Spin Glass model in fig 2.). Communication cost as well as error tolerance are investigated as for the Spin glass. Malfunctioning nodes are no longer restrained by desired Multi-fractal behavior. Two principle malfunctions have been modeled: 1) faulty nodes have the same probability of joining the tree or not 2) faulty nodes randomly choose neighbor tree node to attach to.

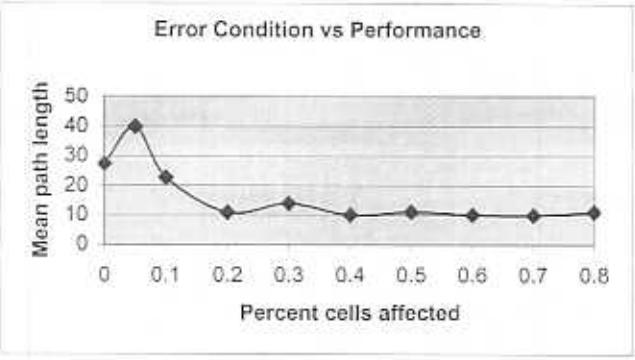

Fig 8 Multi-fractal Effect of error condition on performance

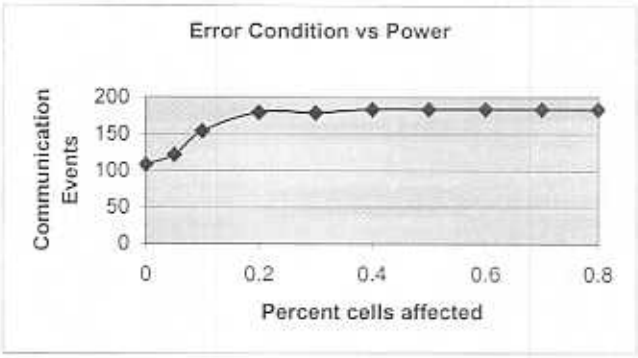

Fig 9 Multi-fractal effect of error condition on power

Fig 8 shows how the error condition affects the routing performance. We notice that there is a slight increase in the mean hops when random error comes into place, and then the mean hops begins to drop down by almost half and stay steadily when error percentage goes up to 20 percent. We may ask does error condition actually improves routing performance? In fact, the shorter mean hops come with the cost of denser trees. Recall that we want a sparse tree, which covers most of the

region. However, the final tree consists of approximately 75 percent more tree nodes compared with the original zero-error tree. The fitness of error conditioned tree is actually not ameliorated.

We also investigate how error condition impacts the communication cost. Fig 9 demonstrates that error condition incurs considerably larger communication events; nevertheless, the extra communication cost is comparably much lower than that of Spin Glass model under the same error circumstance. It indicates that Multi-fractal model is more error resilient in terms of performance and power.

IV. COULOMBIC MODEL

The Coulombic model is a preprocessing step for use with the pheromone algorithm discussed in section V. The pheromone model requires data sources to be evenly distributed throughout the network to conserve bandwidth. The Coulombic model was been designed to fulfill this task. Initially data packets find the data source nearest to the node they are occupying. Pheromone routing is then used to maintain efficient routes between data sources and sinks. A similar approach applied to sensor node placement for sensing coverage is in [3].

The Coulombic model is roughly based on charged particle interactions defined by Coulomb's Law as stated here:

$$F = q_1 * q_2 / (4\pi \varepsilon_0 d^2), \qquad (2)$$

where F is the amount of force on each particle, q1 is the net charge on particle 1, q2 is the net charge on particle 2, ε0 is the constant permittivity of free space, and d is the distance between the two particles.

We utilize two properties of Coulomb's Law: (*i*) the relationship between distance and force and (*ii*) the independent nature of each pair of particle interactions. The second property allows the approach to rely solely on local information and peer-to-peer interactions.

The mapping of the Coulombic model to our abstract network representation follows. Data sources are particles with equal charge and polarity. Distance is measured as the number of hops between data source pairs. The sensor nodes, called free cells, act as vessels through which force can be transmitted. Since repulsion can attain the goal of evenly distributing data sources, we require only one polarity, and hence there is no concept of attraction in this model.

An example output from the algorithm is provided in Fig. 6. In this example checkered objects are data sources, pink cells are data sinks, grey areas are walls (similar in concept to Fig. 1), and white regions are free space. The magnitude of force is indicated by the intensity of the color blue in the free cells.

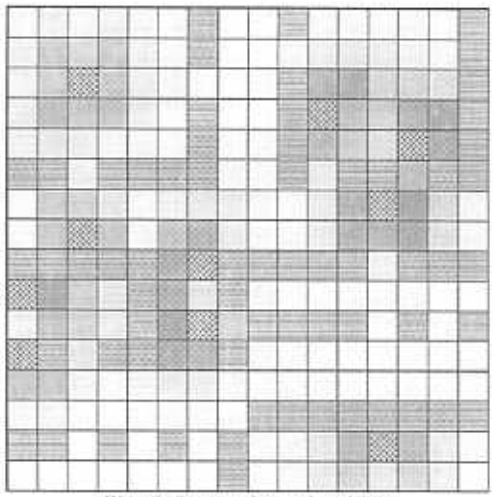

Fig. 6. Output from algorithm

The global behavior of having well distributed data sources emerges from the local node behaviors of all sensor nodes in the network. When examining communication behavior there are three different groupings of cells in the algorithm. Data sources comprise one of these groups. They are responsible for generating force into their neighbors. Free cells form the second group; in essence these cells transmit force in manner similar to Coulomb's Law, namely that the original force generated from the data source dies out with increasing distance. These cells calculate their force by adding a percentage of the force from their neighbors. The third group is composed of walls and other obstacles. These cells do not take part in the transmission of force.

Relying upon local information to track the origin of force without incurring the huge overhead of recording which component of force is due to which data source has caused us to break the force down into a horizontal and a vertical component each of which has a polarity. The force generated from a data source upon its neighboring cells depends upon their relative orientation. Fig. 7 provides a pictorial representation of how the algorithm implements this idea. To determine the net force upon a particular cell it adds a percentage of the force from its neighbors keeping the horizontal and vertical components separated. This allows the force from multiple data sources to merge together and cancel each other out, and hence the routing data is greatly simplified.

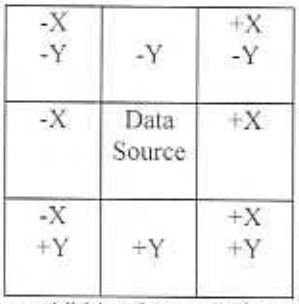

Fig. 7. Data source exhibiting force upon its neighbors (Change)

How the desired behavior of having the data sources move away from each other may not be immediately obvious and requires further explanation. For a data source to be in equilibrium (i.e. it is sufficiently far away from any other data source) the net force that is calculated from adding up the force values from its neighbors will be close to zero. If there is another data source nearby then the aggregate force will not be close to zero, and the data source will use this information to move away from the other data source. As an example consider the two data sources in Fig. 6 that are in the upper right corner. For the data source on the right observe how the force on immediately on its left is much less than the force on the right. This is because the right data source is generating a negative horizontal component in its left neighbor, while the left data source is generating a positive component in this cell, causing some cancellation of force. The result is that the net force acting upon the right data source will have a positive horizontal component. The right data source will then determine that it should move to the right.

This algorithm must assure that the speed at which force propagates through the region is significantly faster than the rate at which the data sources can move, to ensure that data sources are using steady state information. This requirement was derived after observing that it required n generations for the force generated by the data sources to cover an n*n region of space. The algorithm performs this by requiring data sources to wait n generations in between successive moves.

Since the notion of path length is meaningless within the context of this model a different metric than what has been used with the other models discussed within this paper was developed to characterize the performance of the Coulombic model. We define this metric as the mean distance of all of the free cells in a particular scenario from their closest data source. Fig. 8 shows this metric for various parameter sets. The DR (TR) parameter controls how rapidly forces dissipate (diffuse) in the system for each generation of the algorithm. Parameter settings indicated with squares (triangles and x's) are optimal (suboptimal).

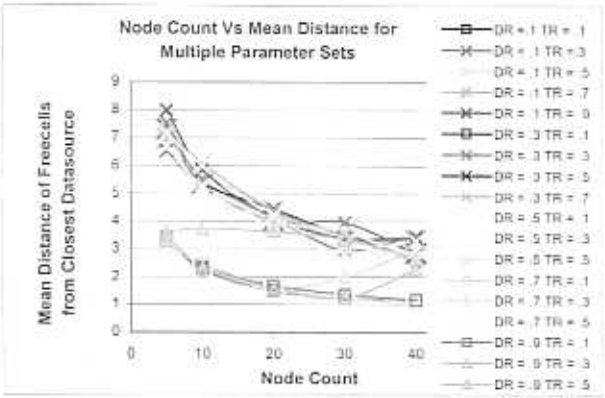

Fig. 8. Performance for various parameters

To guarantee that the Coulombic model converges to a steady state and data sources do not oscillate ceaselessly, the charge diminish rate parameter was introduced. Because of this parameter the charge associated with each data source decreases with each time step. After enough generations have passed the charge of each data source will have reached zero and steady state behavior can be guaranteed. The occurrence of steady state can be predicted using the following equation:

$$N = \ln(T_s) / -CDR, \tag{3}$$

where N is the number of generations needed to attain steady state, Ts is the Stopping Threshold parameter explained subsequently, and CDR is the Charge Diminish Rate parameter. The purpose of the stopping threshold parameter is to determine when the charge on a data source is considered to be zero (i.e. fully diminished). The effects of the charge diminish rate as well as the stopping threshold parameters are illustrated in Fig. 9. This graph indicates that the choice of the charge diminish rate should be higher than .01 for the algorithm to finish in a timely manner, and that the choice of a stopping threshold is of minor consequence in comparison to the choice of charge diminish rate.

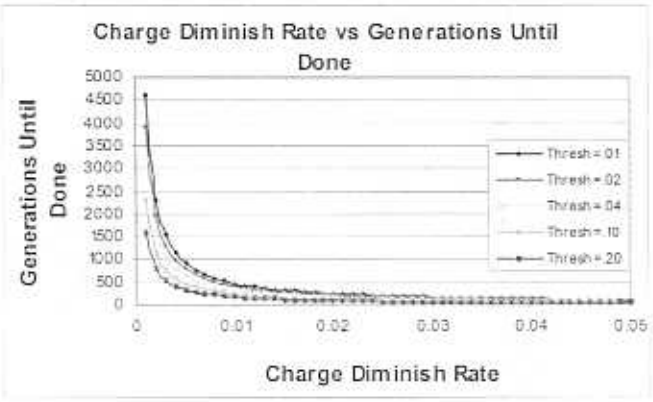

Fig. 9. Parameter effect upon power

Not surprisingly the charge diminish rate parameter has direct control over the power consumption of the model. This relationship is illustrated in Fig. 10. This parameter also has an impact on performance as Fig. 11 indicates. Our results indicate that for this model performance and power have an antagonistic relationship. A low (high) charge diminish rate yields high (low) performance and high (low) power consumption. It is important to note that while a charge diminish rate of zero would yield optimal performance, however the model would not be guaranteed to ever reach a steady state and hence power consumption would be unbounded. Increasing this parameter beyond .08 changes both power and performance by very small amounts. As a result a good parameter setting for both power and performance metrics is in the vicinity of .04.

The pheromone model used is based on how ants forage for food and is related to the routing algorithm in [4]. Data sources (sinks) are modeled as ant nests (food). Messages are ants. The pheromone that the ants release is stored in the sensor nodes distributed throughout the network. Ants attempt to find paths between the nests and food sources. They release two different pheromones: (*i*) search pheromone when they look for food and (*ii*) return pheromone when they have food and return to the nest.

There are two mechanisms for ant movement. The first is that they follow a random walk. The second is that they search for the opposite pheromone of the one they currently release. Ants searching for food tend to follow the highest concentration of return pheromone. Ants returning to the nest tend to follow the highest concentration of search pheromone. This is modeled as a probability distribution where each ant is more likely to move following the pheromone gradient rather than randomly.

The approach in [4] was designed for wired networks. Our scenario is more similar to an open field. In our initial implementation a pathology was noticed where ants moving to and from the data sink would cluster together and never reach their destinations. To counteract this, we caused the ants to be repulsed by the pheromone they currently emit. A parameter was created denoting the relative strength of repulsion and attraction. This compels ants not to stay in one area and solved the pathology.

To evaluate this algorithm's performance we measure the number of hops an ant needs to make a round trip from its nest to a food source. Fig. 13 plots this metric versus the repulsion ratio. A ratio of approximately 90% works best. To evaluate the algorithm's power consumption we have used the same metric as was used in previous sections, namely how often each cell changes its state. Fig. 14 shows how varying the repulsion ratio effects the power consumption of the algorithm. Analysis of this graph indicates that increasing the repulsion ratio beyond 90% has a positive effect on power and that a ratio of 90% (which yields optimal performance) also yields the highest power consumption.

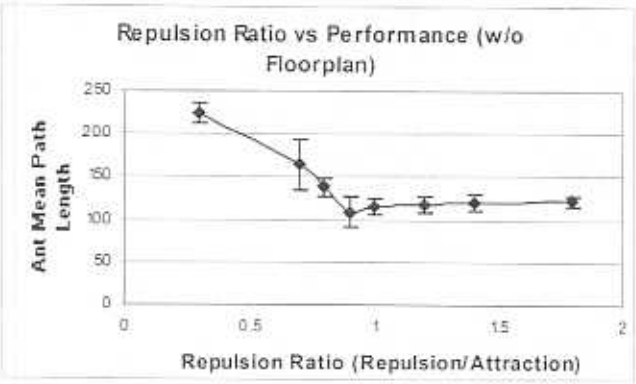

Fig. 13. Effect of repulsion on performance

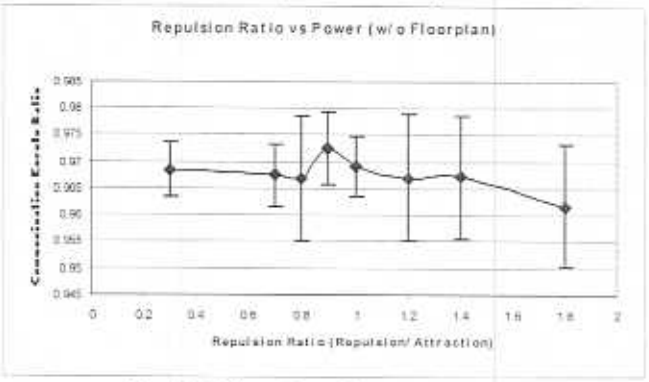

Fig. 14. Effect of repulsion on power

The rate at which a data source (ant nest) generates ants is controlled by a parameter setting. This parameter denotes the probability that a data source will spawn an ant in any particular generation. Fig. 15 shows how varying this parameter affects performance; a spawn frequency at or above 25% yields good performance. We have also evaluated how spawn frequency affects power as is seen in Fig. 16. Power increases quite rapidly for spawn frequencies below 25%. Beyond this point the rate that power increases becomes quite small.

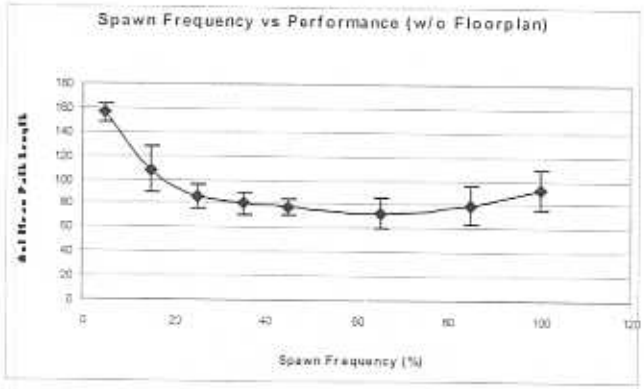

Fig. 15. Effect of spawn frequency on performance

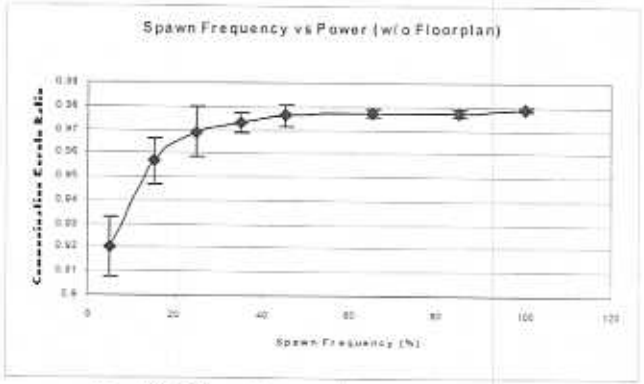

Fig. 16. Effect of spawn frequency on power

We have evaluated this algorithm for several different error conditions to determine its robustness. These error conditions are: *(i)* Haywire Random (a random selection of ants will move randomly for a random amount of time), *(ii)* Haywire Weighted (a random selection of ants will follow the opposite pheromone), and *(iii)* Haywire Cells (a random selection of cells will produce a random amount of pheromone). How these error conditions affect performance is illustrated in Fig. 17. Notice how up to 50% of ants affected with haywire random ends up increasing performance. The effects of haywire cells and haywire weighted are pretty similar, in that they reduce performance pretty drastically up to about 25%, where further hits to performance begin to level off. In Fig. 18 we show how these error conditions affect the power consumption of the algorithm. The effect of haywire random on power is pretty minimal. There is a slight decrease in power around 50% for this parameter. The power consumption for haywire cells is quite dramatic, anything beyond 25% is near maximum power consumption. The decrease in power consumption attributed to haywire weighted can be attributed to how this behavior tends to cause the ants to conglomerate into groups and not move, which has the side effect of reducing communication.

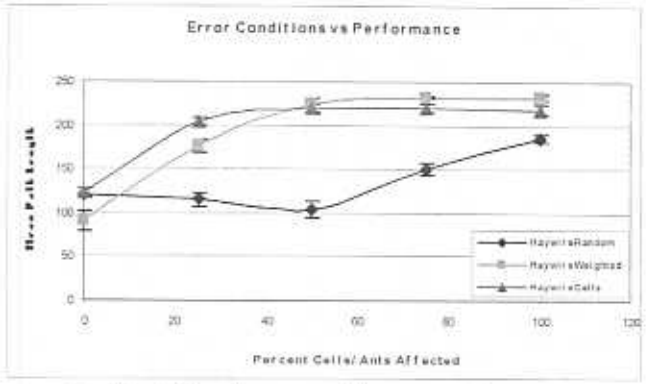

Fig. 17. Effect of error conditions on performance

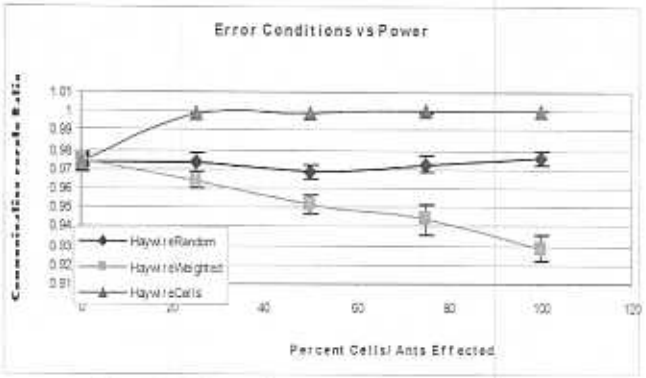

Fig. 18. Effect of error conditions on power

It has been our experience that incorporating a random component into these algorithms usually results in improved performance. The manner in which we have included randomness into this algorithm is by having the ants occasionally move random instead of utilizing the pheromones. The performance impact of this method is presented in Fig. 19. We have determined that a parameter value of about 25% is best for improving performance. We have also evaluated the random movement capability on power consumption. The results are provided in Fig. 20. We have determined that power consumption and the random movement parameter are unrelated.

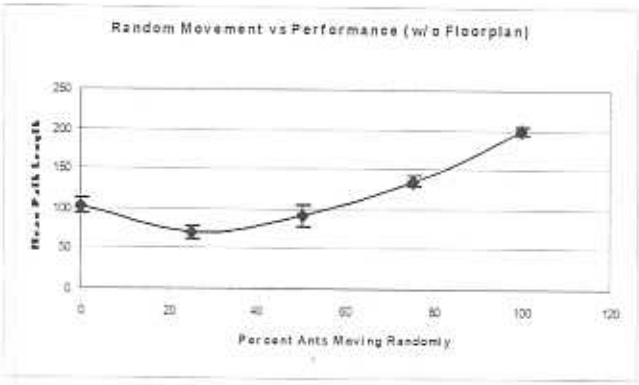

Fig. 19. Effect of random ant movement on performance

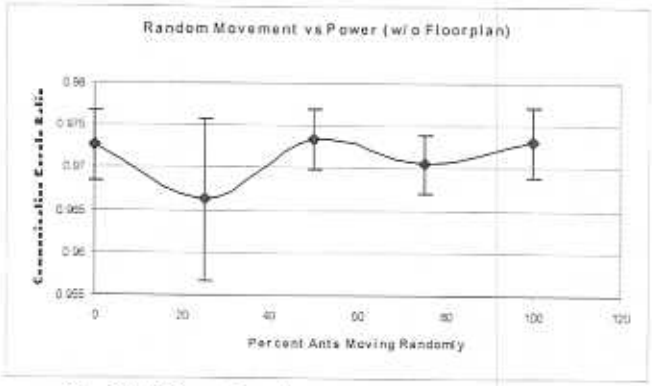

Fig. 20. Effect of random ant movement on power

A parameter known as diffusion rate has been concocted so that the algorithm can control how readily the pheromone in one cell spreads to its neighboring cells. This parameter's effect upon performance is provided in Fig. 21. From our experience a setting of .1 works best. The effect of diffusion upon power is illustrated with Fig. 22. This graph indicates that a diffusion rate of below .2 is good for power.

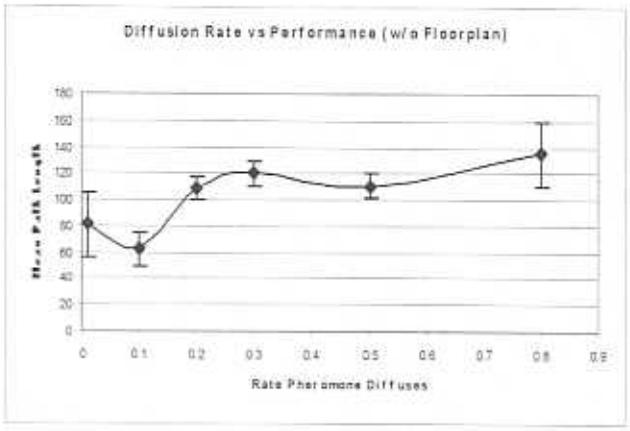

Fig. 21. Effect of diffusion on performance

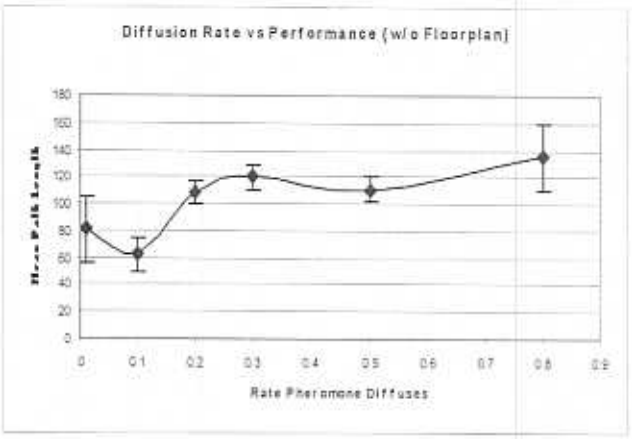

Fig. 22. Effect of diffusion on power

We have included another method to help increase performance. This method is known as evaporation. In each generation of the algorithm a certain percentage of the ant pheromone evaporates (i.e. it disappears). The purpose of developing this technique was an attempt to reduce the power of the algorithm. As Fig. 23 indicates evaporation beyond .05 begins to adversely affect performance. Interestingly enough parameter settings below .05 improve the algorithm's performance.

We have analyzed evaporation based on power as well. Not surprisingly increasing the value of this parameter helps to reduce the power of this algorithm. Unfortunately, any parameter setting that would result in significant power savings coincides with greatly decreased performance. As a result we find a setting of .01 to be optimal.

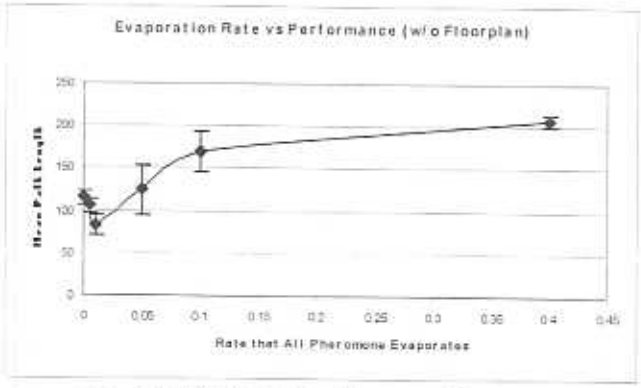

Fig. 23. Effect of evaporation on performance

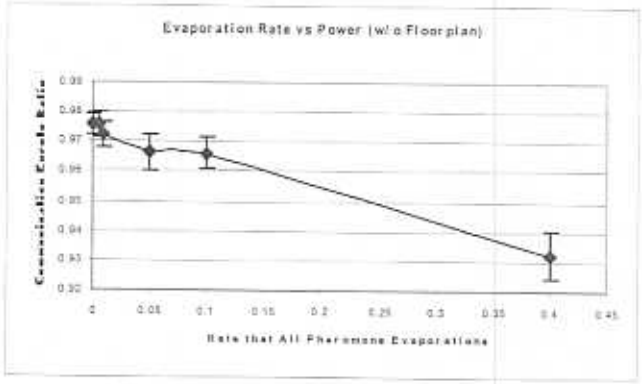

Fig. 24. Effect of evaporation on power

VI. PROTOCOL COMPARISON AND DISCUSSION

Many routing protocols have been proposed for WSNs. The Link State (LS) routing algorithm requires global knowledge about the network. Global routing protocols suffer serious scalability problem as network size increases [6]. Destination-Sequenced Distance-Vector algorithm (DSDV) is an iterative, table-driven, and distributed routing scheme that stores the next hop and number of hops for each reachable node. The routing table storage requirement and periodic broadcasting are the two main drawbacks to this protocol.[6] In Dynamic Source Routing Protocol (DSR), a complete record of traversed cells is required to be carried by each data packet. Although no up to date routing information is maintained in the intermediate nodes' routing table, the complete cell record carried by each packet imposes storage and bandwidth problems. Ad-Hoc On Demand Distance Vector Routing Algorithm (AODV) alleviates the overhead problem in DSR by dynamically establishing route table entries at intermediate nodes, but symmetric links are required by AODV. Cluster-head Gateway Switch Routing (CGSR) use DSDV as the underlying routing scheme to hierarchically address the network. Cluster head and gateway cells are subject to higher communication and computation burden and their failure can greatly deteriorate our system [5]. Greedy Perimeter Stateless routing algorithm (GPSR) claims to be highly efficient in table storage and communication overhead. However, it heavily relies on the self-describing geographic position, which may not be available under most conditions. In addition, the greedy forwarding mechanism may prohibit a valid path to be discovered if some detouring is necessary [7].

The Spin Glass and Multi-fractal models are related to the table-driven routing protocols by establishing routes from every cell to data sink(s). These protocols ensure timely data transmission on demand without searching for the route each time. The Ant Pheromone model is related to the packet-driven protocols. Ants can be viewed as packets traversing from data sources to data sinks. All of the models we presented are decentralized, using only local knowledge at each node. They dynamically adapt to topological disturbances (path loss). Storage requirements for the routing table of Spin Glass and Multi-fractal are low compared with most other protocols, while the Ant Pheromone's storage requirements are even lower than these two.

The Temporally Ordered Routing Algorithm (TORA) is a source initiated and distributed routing scheme that shares some properties with the Spin Glass model. It establishes an acyclic graph using height metric relative to the data sink and also has local reaction to topological disturbances [5].

The kinetic factor in our Spin Glass model and the frequency of ant generation in the Ant Pheromone model provides the system with flexibility in controlling routing behaviors under various conditions. Route maintenance overhead is moderately high for the Spin Glass model.

The Multi-fractal approach, as a probabilistic space-filling curve, has very light computation and communication load, and overhead is saved in route discovery and maintenance. This is at the cost of a higher distance to the data sink(s). Route maintenance overhead for the pheromones is low due to the reduced number of nodes involved in each path. Since the Multi-fractal model strives to cover the sensor field by using as few cells as possible, the sparse routing tree sparse conserves energy. The shortest routes to the data sink are not found using the Multi-fractal model.

On the other hand, Spin Glass model is more sensitive to internal errors since any possible error may diffuse throughout the network. The Multi-fractal and Ant Pheromone models are very resistant to internal errors. The time required for the Ant Pheromone algorithm to converge to a steady state is much longer than required by the other two adaptations. For applications requiring short data paths, the Spin Glass model is preferred. For overhead sensitive applications that require quick deployment, the Multi-fractal model is a better candidate. If error resilience and low overhead are the

principle requirements, then the Ant Pheromone model is appropriate. Hybrid methods or switching between methods at different phases may be useful.

VII. CONCLUSION

The purpose of our work is to develop adaptive networking methods for ad hoc WSNs. We performed analyzed the algorithms based on resource consumption, fault tolerance, number of nodes required, sensitivity to algorithm parameters, and critical points where phase changes occur.

This conference paper summarized some of our results. We are now using these insights to design wireless routing protocols in conjunction with researchers at the University of Wisconsin. Two applications are foreseen for these adaptive protocols.

One application is for the system to tolerate intermittent hibernation by a non-negligible subset of the WSN nodes. This should significantly prolong the lifetime of the system.

The other application is to maintain multiple routes to a single data sink. This should both prolong the system lifetime and support information assurance requirements.

In addition to this, we are continuing our analysis of system adaptation. A unifying abstraction is being considered that contains these approaches as a subset. It may then be possible to analytically derive local behaviors to maintain globally desirable system attributes.

Our approach is to consider and test the adaptation problems of the system at an abstract level first. Insights gained at this level can then be used in protocol design and implementation. We are currently designing the protocols for implementation and testing with standard network tools like NS-2.

REFERENCES

1. K. Chang, C. Chonge, and Y. Bar-Shalom, "Joint Probabilistic Data Association in Distributed Sensor Networks," *IEEE Transactions on Automatic Control*, vol. AC-31, pp. 889-897, 1986.
2. R. R. Brooks, P. Ramanathan, and A. Sayeed, "Distributed Target Tracking and Classification in Sensor Networks," *Proceedings of the IEEE*, Invited Paper, Accepted for Publication, February 2003.
3. Y. Zou and K. Chakrabarty, "Sensor Deployment and Target Localization Based on Virtual Forces." *IEEE Infocom Conference, 2003.*
4. Marco Dorigo, Vittorio Maniezzo, Alberto Colorni, "The Ant System: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics Part B.* 26(1):29-41, 1996
5. Elizabeth M. Royer. A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks *IEEE Personal Communication* April 1999.
6. James F. Kurose and Keith W. Ross Computer Networking a Top-Down Approach Featureing the Internet AW Higher Education Group 2003.
7. Brad Karp and H.T.K Ung Greedy Perimeter Stateless Routing for Wireless Networks *Proc. of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking 2000*
8. Tom Gottschalk, Dan Davis Hrothgar Project Center for Advanced Computing Research at Caltech 2000
9. A. Major, ASA, MAAA and Yakov Lantsman. Part 1 Actuarial Application of Multifractal Modelling
10. Richard J.Gaylord Kazume Nishidate Modeling Nature Cellular Automata Simulations with Mathematica