Geodesic packet routing over self-organizing mobile ad-hoc virtual coordinates

Thayne R. Coffman
Christine Julien

TR-UTEDGE-2006-011

Geodesic packet routing over self-organizing mobile ad-hoc virtual coordinates

Thayne R. Coffman and Christine Julien, *Member, IEEE*

***Abstract*—The development of routing protocols for mobile ad hoc networks is a challenging and active research area, and a variety of approaches have been developed. Among them, geographic routing strategies use knowledge of the devices' physical positions to achieve efficient routing performance – a critical goal for networks of resource-constrained devices. Strategies have also been developed to perform geographic routing when the physical locations of the nodes are unknown. Many approaches accomplish this by generating virtual coordinates. We propose a new approach for generating virtual coordinates that is inspired by the Kohonen self-organizing feature map (SOM), a neural network approach to unsupervised learning. The self-organizing map has attractive properties including parallelizable training and resilience to changes in its topology. We present our approach and the results of some basic simulations that measure its ability to perform efficient geodesic packet forwarding over its virtual coordinates.**

***Index Terms*—communication system routing, mobile networks, routing, self-organizing feature maps**

I. Introduction

Mobile ad hoc networks (MANETs) are an emerging computing paradigm, enabled by recent advancements in processing capabilities, wireless networking, and power reduction. MANETs provide the ability for handheld or independent mobile devices to self-organize and collaborate without any prior infrastructure. They have many military and commercial applications and have unique characteristics when compared to traditional networks. While advances occur daily, modern mobile ad hoc networks are still typically populated by devices with severe constraints on processing power, battery life, and bandwidth.

Manuscript received December 6, 2006.

T. R. Coffman is an employee of 21st Century Technologies, 4515 Seton Center Parkway, Suite 320, Austin, TX, 78759 (phone:512-342-0010, fax: 512-342-0196, e-mail: tcoffman@21technologies.com) and a Ph.D. student in the Department of Electrical and Computer Engineering at the University of Texas at Austin.

C. Julien is an Assistant Professor in the Department of Electrical and Computer Engineering at The University of Texas at Austin (e-mail: c.julien@mail.utexas.edu).

MANETs achieve infrastructure-free operation, in part, by having each device forward packets for other devices. The strategy used for forwarding is set by the network's routing protocol. The routing protocol performs multi-hop packet forwarding to let devices exchange information even when outside of each other's radio transmission ranges. Developing routing algorithms for MANETs is more difficult than for traditional networks because the devices can move or unexpectedly stop operation. At the same time, the prevalence of severe resource constraints makes it even more critical that the routing algorithm performs efficiently. Routing packets inefficiently wastes battery power, processing power, and bandwidth [1].

Virtually every mobile ad hoc network would benefit from improved routing performance. Most of the original MANET applications were for military communication on a battlefield, but commercial applications are increasing. Commercial applications include support for collaboration at conferences, "smart" classrooms, surveillance, and environmental monitoring.

In this paper we propose and evaluate a new approach to performing geographic routing in mobile ad hoc networks when device locations are unknown. We present a method for generating virtual coordinates that is inspired by Kohonen self-organizing feature maps (SOMs). The approach benefits from SOMs' ability to spontaneously determine useful coordinate values for lattices of nodes in a network when presented with training data, as well as their resilience to changes in network topology. It has a number of attractive properties for use in MANETs, and holds the promise of low-overhead collaborative generation of virtual coordinates.

Following our introduction, Section II describes related work, Sections III and IV present our approach and evaluation methodology, respectively, and results are presented in Section V. Section VI discusses the results and ways to address common MANET issues, and Section VII concludes the paper.

II. Related Work

Because we are proposing a new approach for enabling the use of geographic routing strategies when exact device positions are unknown, existing ad hoc routing protocols are of interest. Of more specific interest are existing geographic routing protocols, and most directly relevant are other techniques that enable the use of geographic routing when exact device locations are unknown.

As stated above, one primary focus of an ad hoc routing protocol is to define the strategy for multi-hop packet forwarding, which enables interaction among devices that are not within direct communication range. Ad hoc routing protocols can generally be categorized into two main groups – table-based protocols and on-demand protocols. Table-based (or proactive) protocols maintain routing information for all nodes in the network, in anticipation of potential communication requests. They tend to have low initial latency when new combinations of nodes start communicating, and they require no special route establishment overhead. However, they can require the storage of large amounts of information on each node, and as a result they can have poor scaling properties [16]. In contrast, on-demand or reactive protocols (including AODV, DSR, and TORA) only maintain routing information for pairs of devices that are actively communicating [16]. This reduces the amount of information that needs to be stored on each node. Reactive protocols typically have nonzero channel setup costs and latency at the beginning of an interaction, but are generally considered more scalable than proactive protocols. For this reason, they are more widely used in mobile ad hoc networks than proactive protocols.

One particularly interesting proactive approach is described in [10]. Their confidence-based dual reinforcement Q-routing (CDRQ-routing) approach is inspired by a combination of reinforcement learning and dynamic programming. Each device X builds its best estimate, $Q_X(Y,Z)$, of the cost of sending a packet to destination Y through its direct neighbor Z. The reinforcement learning strategy continually refines each device's estimates $Q_X(Y,Z)$, based on neighboring devices' estimates $Q_Y(W,Z)$ and timing information recorded from actual packet deliveries. This lets CDRQ-routing adapt to changes in network topology as well as changes in local traffic load on various parts of the network. This approach has similarities to self-organizing map training, which is central to our approach. However, CDRQ-routing seems to suffer from the scalability problems typical of table-based protocols, in that large amounts of information must be maintained at each node.

A. Topology Discovery

In non-geographic routing protocols, routing decisions are typically based on the connection topology. In traditional networks, this topology is relatively well-known, and prior knowledge can be used in establishing routing rules. In MANETs, however, topology is generally unknown. As a result, topology discovery techniques are used to support the use of non-geographic routing protocols on mobile ad hoc networks.

In [4], the authors present a topology discovery algorithm called TopDisc. TopDisc is intended for use on sensor networks and operates on the connection topology directly without any notion of physical sensor location. It is a hierarchical approach where sensor nodes self-organize into a tree of clusters. TopDisc requires only the use of local information for setup (one packet per node), and only a subset (a dominating set) of the sensors needs to transmit that packet during the setup process. TopDisc provides an efficient non-geographic route setup strategy and routing protocol for sensor networks.

In [5], Haas describes the Zone Routing Protocol (ZRP), a hybrid proactive/reactive approach intended for use in ad hoc networks with high degrees of mobility. ZRP maintains up-to-date topological maps of a zone centered on each device and uses this to reduce the inefficiency problems found in both proactive and reactive protocols under high mobility. Even though the network is divided into zones, ZRP is a flat protocol – these zones overlap and are not organized into a hierarchy.

B. Geographic Routing

Geographic routing protocols use information about the physical positions of each device to route packets very efficiently. Geographic routing protocols can be either proactive or reactive, but tend to be flat instead of hierarchical. In [7], the authors give an extensive survey of current geographic routing protocols – in this paper we simply discuss some major issues and present a few approaches that we find relevant.

In the simplest and earliest form of geographic routing, called geodesic packet forwarding or greedy routing, each packet is forwarded at each hop to the immediate neighbor located closest to its destination [3]. Under this strategy, it is possible for a packet to be routed to a local minimum in the proximity of its destination (a dead end, also called a routing failure), and so some recovery strategy must be introduced to

deliver these stranded packets [9]. One common recovery mechanism is called perimeter routing (closely related to face routing [9]), and involves a "right-hand-rule" search around the perimeter of the geographic destination [7]. Because recovery mechanisms tend to be much less efficient than the primary geographic routing protocol, one measure of the performance of a geographic routing protocol is its success rate – the percentage of packets that arrive at their destination without requiring invocation of the recovery mechanism. We use this interpretation of success rate as the primary metric in our simulations described below.

A seminal geographic approach, GOAFR+ (read as "gopher plus") is presented in [9]. The authors find it to be "outstandingly efficient" in the average case and have formally shown it to be asymptotically worst-case optimal. GOAFR+ uses local and one-hop neighbor locations to perform greedy routing with face routing as the recovery mechanism.

The directed diffusion approach presented in [8] is tailored for sensor networks and establishes gradients for each data request. Their communication model is data-centric, in that all non-overhead communication is assumed to be either a request for or a piece of data. Data packets always carry a set of attribute-value pairs, and those attributes may correspond to either physical locations or data content. Communication is initiated by a device that registers interest in a set of named data. This interest implicitly sets up a gradient in the system for matching data. These gradients are used to route the responses, which "sink" towards the interested device.

C. Geographic Routing Without Locations

Establishing node positions with localization equipment like global positioning systems (GPS) can be too resource-intensive for some ad-hoc networks. Even if resource conservation is not an issue, GPS signals may be unavailable for other reasons, e.g., indoor operation or intentional jamming by an adversary. It is reasonable to want to achieve the efficiency of geographic routing protocols even in situations where the devices on the network cannot determine their true locations.

In these situations, the question becomes how best to estimate a set of virtual coordinates that can be used in the geographic protocols as substitutes for the devices' true locations. A number of groups have proposed ways to generate these virtual coordinates. Some approaches attempt to accurately estimate the true device positions. Other approaches, however, generate a set of virtual coordinates that provides efficient routing behavior without necessarily corresponding to physical locations.

The Self-Positioning Algorithm (SPA) described in [3] uses estimates of the distances between devices to build a relative coordinate system in two dimensions on which each device's position is determined. Inter-device distances are measured based on time-of-arrival (or equivalently, time-of-flight) computations. Through this strategy, devices can determine their positions in the virtual coordinate system using only local information.

In [12], the authors claim to present "the first approximation algorithm" for generating virtual coordinates from connectivity information alone. The main contribution of the paper is a formal proof of the accuracy level of their approach. They achieve this by defining a metric on the network graph and formulating the approach as a graph embedding optimization problem solved via linear programming.

In [13], the author develops a global coordinate system from local information and communication only. The approach is inspired by biological systems that use chemical gradients to determine the positions of cells within an organism. Two important findings are presented. First, there is a fundamental limit on the resolution of any coordinate system determined strictly from local communication, and second, a random distribution of devices allows the approach to achieve more accurate coordinate estimates than a regular grid.

The authors in [17] present an $O(n^3)$ algorithm for deriving location estimates solely from connectivity information based on multi-dimensional scaling. When additional information is available, the approach can also incorporate information on estimated inter-device distances or known positions for some subset of the devices.

A group at U.C. Berkeley generates virtual coordinates through an iterative relaxation scheme [15]. The heart of this iterative approach is very similar to some used for visual graph layouts in graphical user interfaces. Devices on the geographic perimeter of the network estimate their own locations by another means, and those estimates are then held constant. On each iteration, non-perimeter devices update their self-localization estimates to be the mean of the estimates of their immediate neighbors. The basic approach requires devices on the physical perimeter of the network to know that they are on the perimeter and to know their true locations. The work then builds on the basic approach to eliminate the need for perimeter devices to know their positions, and finally to eliminate the need for perimeter devices to even know that they are on the perimeter. They achieve good routing success rates and eliminate the need for any prior knowledge about the positions of the devices, but their process of negotiating

between the devices in the network to collectively determine which are on the perimeter introduces considerable setup overhead.

D. Self-Organizing Maps

Kohonen self-organizing feature maps, also called simply self-organizing maps or SOMs, are a type of neural network used for unsupervised machine learning. They have been used in many cases for clustering data, estimating probability density functions (PDFs) from a data sample, or reducing the dimensionality of a dataset. A full introduction to self-organizing maps is beyond the scope of this paper, but can be found in [6]. We instead attempt to highlight only the most relevant information.

Self-organizing maps are built from a lattice of nodes, usually with a regular topology. The training process for a SOM is based on competitive and cooperative interactions between nodes in the lattice. Each node has an associated archetype location in the input space. During training, samples are presented from the input space in sequence, and nodes compete to "win" each new training sample. The winner is the node whose archetype is closest to the training sample. The winning node is then modified such that its archetype location moves towards the training sample it has just won. In addition to its motion towards the training sample, the winning node's neighbors in the lattice move towards the training sample as well (although they move a shorter distance). Thus the nodes learn cooperatively.

An excellent description of the effects of this strategy is given in [6]. In training, each neuron becomes selectively tuned to a different input pattern or set of input patterns. The locations of those neurons spontaneously organize towards a topological ordering by which meaningful coordinates can be assigned to a given input pattern. These coordinates capture intrinsic statistical features of the input patterns, and the spontaneous organization of the coordinates drives the name "self-organizing map."

SOM training has a number of interesting characteristics. Through the course of training, node interactions with their neighbors become both weaker and more local over time to ensure that the map converges. Training typically divides (by internal dynamics, not by human coercion) into an organization phase followed by a convergence phase. The initial organization phase is almost always shorter than the convergence phase by factors as large as 10-100. Many parallelized implementations of SOM training exist, demonstrating that SOM training can be effectively distributed. A sequence of characteristic SOM training states is shown for intuitive understanding in Fig. 1.

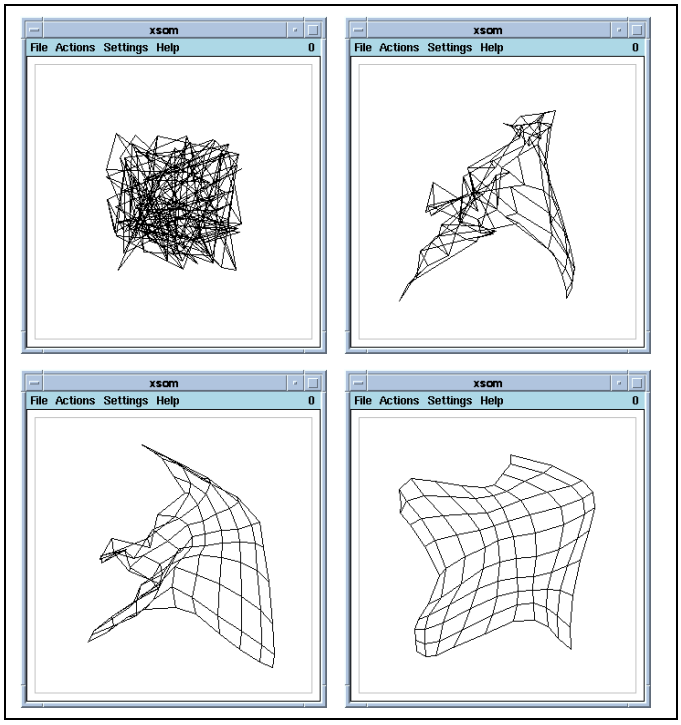

Fig. 1. Example SOM organization during training: (top left) 0 training iterations, (top right) 20 iterations, (bottom left) 40 iterations, (bottom right) 80 iterations [2]. Note the organization of node coordinates from randomness to topological ordering.

III. Self-Organizing Mobile Ad-hoc Virtual Coordinates

Our goal is to generate virtual coordinates for each mobile host that will enable geodesic packet forwarding for efficient multi-hop routing. We want to achieve this with a minimal amount of centralization and communication overhead.

One strategy for generating virtual coordinates is to try to estimate coordinates that are very close to the devices' true physical locations. That is, however, a stronger requirement than we need. A more attainable but equally effective strategy is to generate, for a given node N, virtual coordinates that are close to its neighbors' virtual coordinates and far from the coordinates of all of its non-neighbors. These coordinates may not reflect the devices' true physical locations, but they can still be effective for use with geodesic packet forwarding. Generating these coordinates is an excellent application for self-organizing maps, and we call the result Self-Organizing Mobile Ad-hoc virtual coordinates, or *SOMA coordinates*.

Typical self-organizing maps use a regular two-dimensional lattice of nodes, where all nodes have identical connectivity except for those on the perimeter, as was shown in Fig. 1. SOMs with six neighbors per

node instead of four are also often used.

In contrast with these regular lattices, we use the irregular topology of our mobile network to define the connectivity of the nodes in our lattice. It is trivial for each node in the network to discover its set of neighbors via heartbeat messages or other means. We will postulate a self-organizing map lattice with exactly the same topology as our mobile network.

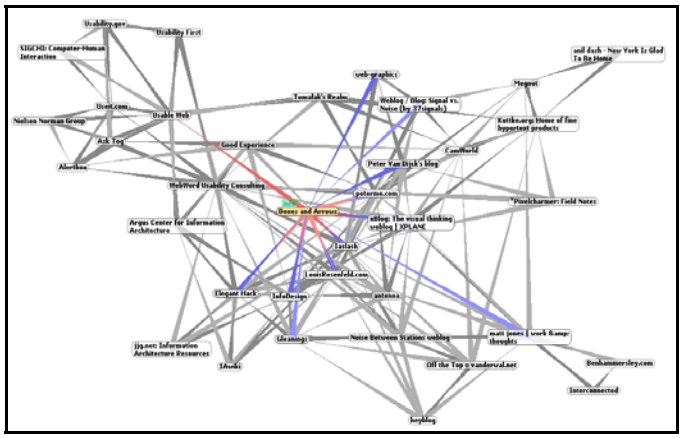

Fig. 2. One possible result of virtual position layout on an irregular network topology [18]. Node positions do not necessarily correspond to physical locations, but are instead dictated by the local connectivity with neighboring nodes.

It is worth noting that the network topology does not need to be globally known – it is sufficient for nodes to know only their set of immediate neighbors. Contrast the regular topology of Fig. 1 with that of Fig. 2, which shows one potential SOM topology and layout derived only from direct neighbor connectivity information. Fig. 2 illustrates what a set of virtual coordinates might look like after training. On initialization they will look random, similar to the top left image in Fig. 1.

In typical SOM training with a regular lattice, the nodes spread out towards a "smooth" representation of the statistics of the training data. When trained on uniformly distributed 2-D data, the virtual coordinates for our irregular lattice will instead spread out towards a state where neighbors tend to have similar virtual coordinates and non-neighbors tend to have dissimilar virtual coordinates. These coordinates may or may not reflect the devices' true physical positions, but they will still be effective for use in geodesic packet forwarding.

IV. Evaluation Approach

Our evaluation approach is based on empiric simulation of the algorithm routing packets in an instrumented simulation framework. In each simulation, we train the SOMA position estimates and then evaluate their effectiveness for geodesic packet forwarding.

A. Simulation Framework

To test our approach, we developed a simple simulation framework in Java. Following the parameters used in [15], we set up a series of scenarios with 3200 devices distributed uniformly in a 200x200 meter area, with identical transmission ranges of 8m. These parameters result in a very dense network with typical averages of 15.5 neighbors per node, which provides a challenging test for the approach. Each simulation routed 2500 packets with geodesic packet forwarding operating on the devices' virtual coordinates. Node mobility was not considered but there are a number of reasons why the approach should perform well with mobile nodes, as discussed in Section VI. Ten simulations were run on each set of parameters and average results are reported.

The primary performance metric is the percentage of packets that were successfully routed without needing to resort to a recovery mechanism for resolving a dead-end. Note that this is not the percentage of packets that would reach their destination, but rather the percentage of packets that can reach their destination without invoking a costly recovery mechanism.

In our study, SOM training was completed before any packets were routed, and training was done in a centralized manner assuming global knowledge of nodes' virtual coordinates. This training strategy must obviously be extended before being applied to realistic scenarios. Centralized training was used for ease of experimentation and proof of concept. Decentralized and on-the-fly training is feasible, albeit at some cost in overhead. Discussion of why decentralized training is feasible, potential improvements to our simulation framework to model decentralized training, and our reasons for being optimistic about SOMA performance in distributed scenarios are all discussed below.

B. SOM Training

Training self-organizing maps requires the specification of a set of parameters. Typically, training occurs in stages, and parameter values are specified for each stage. The most critical training parameters for each stage include the learning rate, the neighborhood size and function, and the number of training iterations in the stage.

The learning rate and neighborhood function (described below) determine the degree to which nodes move towards a particular training sample. For each sample, the "winning" node – the node that was already closest to the training sample – moves even further towards that training sample.

The learning rate, η, is a number in the range (0, 1]

that determines how far the winning node moves. With a learning rate of 1.0, the winning node's virtual coordinates are simply replaced by those of the training sample. With a learning rate of 0.0, the winning node wouldn't move at all, and with a learning rate of 0.5, the winning node would move halfway to the training sample.

The winning node is not the only node that moves – as it moves, its direct and indirect neighbors move with it. The neighborhood function determines the extent to which direct and indirect neighbors of the winning node are also moved. It is a function of the distance (in terms of hops in the MANET) between the winning node and the indirect neighbor, and it is parameterized by the neighborhood size. The neighborhood function at distance d and neighborhood size σ is typically written as $h(d;\sigma)$. "Rectangular" neighborhood functions are sometimes used, such that

$$h(d;\sigma) = \begin{cases} 1, & (d \le \sigma) \\ 0, & (d > \sigma) \end{cases} \qquad (1)$$

Research has shown, however, that Gaussian neighborhood functions typically yield faster SOM convergence[6]. When using a Gaussian neighborhood function, the neighborhood size is used as the standard deviation of the Gaussian, resulting in the form

$$h(d;\sigma) = \exp\left(-\frac{d^2}{2\sigma^2}\right) \qquad (2)$$

Both the learning rate and the neighborhood size typically decay over the course of a training stage. The number of iterations in a training stage determines both the number of training samples presented in that stage and (indirectly) the rate of decay in the two other parameters. Parameter decay is normally chosen to be either exponential or linear. Exponential decay can improve training success but also introduces more complexity in requiring computation of the exponential decay constant [6].

C. SOMA Training Strategy

In our experiments, we chose an extremely simple training strategy. There was only one training stage. The learning rate did not decay, but was instead fixed at η = 0.2. The neighborhood function was a Gaussian and the neighborhood size decayed linearly. The initial value of the neighborhood size, σ_0, was varied between runs, but the final value was always 2.0. We also truncated our neighborhood at 2σ; even though the Gaussian neighborhood technically goes on to infinity, we considered interactions beyond 2σ negligible. The number of iterations in the single training stage was also varied between runs. Initial neighborhood size was varied between 5 and 12 in increments of 1. The number of training iterations was varied between 100 and 1200 in increments of 100.

V. RESULTS

A baseline simulation for comparison was run using geodesic packet forwarding over the nodes' true positions, with the other parameters described above. The success rate (without resorting to a recovery mechanism) for ten simulations averaged 96.5%. Since it is based on complete knowledge of positions, we took this baseline to be an example of successful routing rates. Other research groups have shown that geodesic packet forwarding over virtual coordinates can outperform true physical positions, a finding reinforced by our results below.

Another ten simulations were run using completely random virtual coordinates, resulting in an average success rate of 0.76%. We treated this as a lower baseline on performance because it was based on completely random guesses for virtual coordinates.

The average success rates for geodesic packet forwarding over SOMA virtual coordinates are shown in Fig. 3. The results are extremely encouraging. The top performance in the average sense was achieved at 1100 training iterations with σ_0 equal to 11, and yielded a success rate of 95.5% – only one percent away from the performance on true physical locations. Of the 96 parameter combinations tested, 9 generated average performance results of over 90%. Not surprisingly, increasing the number of training iterations and increasing the value of σ_0 both tend to improve average success rate. The increase in success rate that comes from increasing each of these parameters is fairly well-behaved and monotonic, which is also encouraging. It is clear from the results, however, that the training parameters must fall in a certain range to achieve a good success rate. We expect that the ideal parameter range will be a function of the number of nodes in the network and/or the average number of neighbors per node.

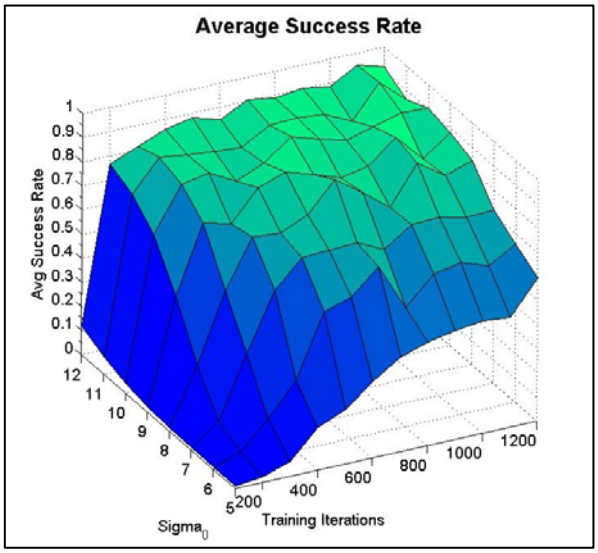

Fig. 3. Average success rate for geodesic packet forwarding over SOMA virtual coordinates. SOMA coordinates achieve high success rates even with moderate numbers of training iterations.

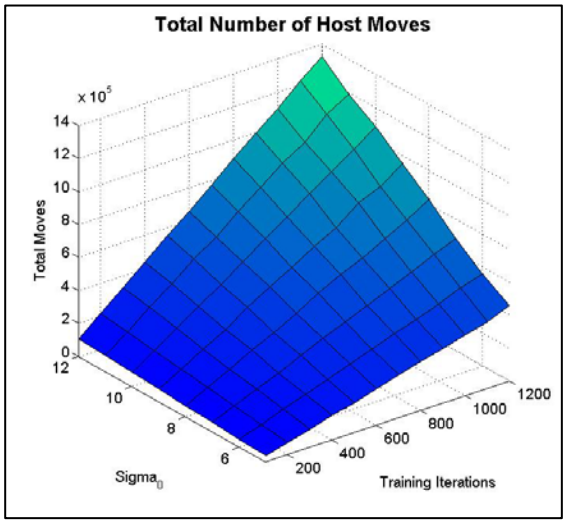

Fig. 4. Total number of adjustments to SOMA virtual coordinates during training. Adjustments increase significantly as training iterations and σ_0 are increased.

Fig. 4 quantifies the price that is paid for virtual coordinate generation. This price is paid in the number of refinements to virtual coordinate estimates that must be made during the training process. For each training sample generated during SOM training, potentially many hosts will be moved. Each of these "host moves" will require at least one communication between a pair of hosts and is more likely to require a communication between the winning node and one of its indirect neighbors. As such, we expect at least one packet worth of overhead traffic to be generated per "host move".

As you can see in Fig. 4, the total number of host moves generated by our choice of training parameters varies between 40,000 and 13,000,000. Needless to say, this is a substantial amount of overhead. Because the average success rate rises faster in certain parameter space regions than the number of host moves rises, we expect to be able to find a "sweet spot" where an acceptable routing performance is achieved with an acceptable overhead. To illustrate, the maximum-overhead test (1200 iterations at an initial neighborhood of 12) achieved a success rate of 91.5%. Using 1000 iterations and an initial neighborhood size of 9, however, SOMA still achieves a success rate of 91.2%. A reduction in overhead by a factor of 16 reduced the average success rate by only 0.03%. We also expect that further reductions in overhead could be achieved by pursuing more refined SOM training strategies.

We see another reason to be optimistic about the success rates that can be achieved by using SOMA virtual coordinates. Fig. 5 shows the percentage of individual simulation runs (out of ten runs under each set of parameters) that generated success rates above 95%. Even parameter settings that generated lower total average success rates still had a significant percentage of runs with success rates over 95%. For example, consider the results with 500 iterations and an initial neighborhood size of 9. Averaging across all ten simulations, SOMA achieved an average success rate of only 83% for its 400,000 host moves. However, of those ten runs, six of them had success rates over 95%. Thus the median success rate was much higher than average.

This holds in general – we typically see a dichotomy of performance when looking at the individual simulations. For a set of training parameters that generate suboptimal performance in the global average, there are still a substantial number of individual runs that generate excellent performance. This is not unexpected in a SOM-based strategy, where a suboptimal choice of training parameters can lead to maps that might (or might not) fall into local maxima, which are often manifested as "knots" in the map. This dichotomy introduces the possibility that we can train with a less expensive set of parameters, rely on our ability to detect poor self-organization by an unacceptable number of routing failures (perhaps more than 5%), and then re-initialize the training if needed.

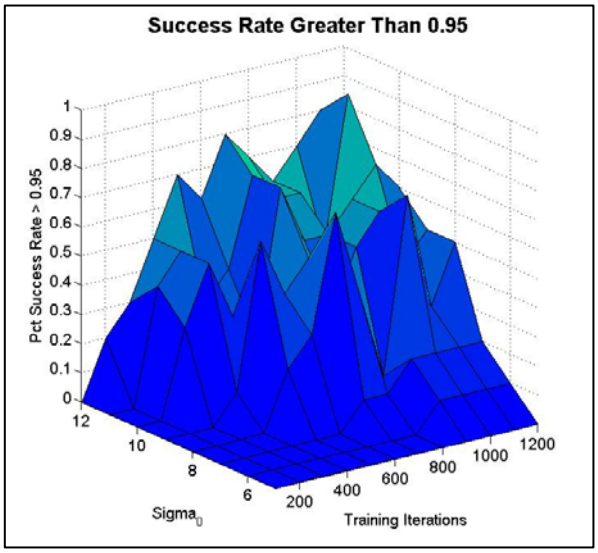

Fig. 5. Percentage of individual simulation runs with success rate above 95%. Average success rates shown in Fig. 3 do not tell the entire story – the bifurcation in observed results leads to large percentages of simulations with high success rates, even for low values of σ_0 and relatively few iterations.

VI. Discussions

SOMA performance in this evaluation is encouraging, and we have reasons to believe that the approach will extend well to evaluations that incorporate other issues in mobile networks. We believe the most interesting areas for further examination are to extend the simulation environment to be more realistic, to decentralize SOMA training, and to explore more complicated training strategies.

We have used a home-grown network simulator to test our approach thus far, but the use of a more widely accepted simulator would enable an apples-to-apples comparison with other approaches, as well as a more realistic simulation environment. We believe the ns-2 simulator, developed by the Information Sciences Institute at the University of Southern California (USC-ISI), would be appropriate. This would allow us to quantitatively compare the amount of overhead our approach imposes with those of other techniques.

In the simulations we have presented, our nodes were assumed to be immobile. Obviously, this restriction misses the point of mobile ad hoc networks. We believe that the SOM approach will fare well under node mobility because both artificial SOMs and the biological SOMs found in the human brain have a well-documented history of showing resilience under changing input statistics and connection topologies [6]. The use of the ns-2 simulator will let us verify this.

We believe that it is less common for a MANET to be simply "turned on" with 3200 nodes active on it, than to have a few nodes present at startup and have nodes added to the network incrementally. In this scenario, we believe that the SOM approach will compare favorably with other approaches and require much less self-organization overhead, again because of SOMs' resilience to changes in topology. Self-organizing a small network and growing it incrementally is much easier than self-organizing 3200 nodes all at once.

We also believe that the SOMA approach (or any approach based on virtual coordinates) will perform better in comparison to using the devices' true physical coordinates when the devices are not uniformly distributed over a physical area. Areas with holes in the middle, blockages, or "dead zones" could present problems with the use of true physical coordinates, but the use of virtual coordinates should circumvent these problems. We would like to study this further.

It is important to decentralize our SOM training. The centralized implementation proves the feasibility of the approach, but the training process must be decentralized to be useful in a deployed system. Additional coordination will be required between nodes, but that coordination is limited by twice the neighborhood size. Generating the training data itself is easily decentralized.

There is no inherent need for a hard boundary between the training stage and an operational stage. Rather, individual nodes could each keep an internal "temperature" value that determined their neighborhood size, learning parameter, and the rate at which they generated random training samples. This temperature could be a function of their observed local routing failure rate and the rate at which they are gaining and losing neighbors. In this manner, the SOMA approach could adapt to changes in local regions, aggressively retraining when needed and minimizing communication overhead when possible.

A significant design space also exists in selecting training parameters and training strategies, including work to automatically determine a set of initial training parameters for a given network. More complicated training strategies have the potential to significantly reduce the amount of communication overhead required for self-organization. We described above how SOM training typically self-divides into a short organization phase and a longer convergence phase. For our purposes, training simply to the completion of the organization phase should still result in high success rates. Techniques to track the coordinates' transition between organization and convergence would be of interest, as would techniques to detect poor self-organization and restart training. We believe both of these to be feasible, as nodes should be able to easily track their individual routing failure rates and trends in those rates.

In the study presented, our primary performance metric was the routing success rate – the percentage of packets delivered without needing to invoke an expensive recovery mechanism. Based on our assertion that "virtual coordinates should be similar for neighbors and dissimilar for non-neighbors," we believe that some of the metrics used in clustering algorithms to measure within-cluster and between-cluster variance will also be good measures of the quality of virtual coordinates. To a first approximation, we found that the ratio of average between-neighbor distance to average between-non-neighbor distance seem correlated with routing success rates.

VII. CONCLUSION

The self-organizing mobile ad-hoc (SOMA) virtual coordinate generation approach is a new method for enabling geographic routing in mobile ad-hoc networks when the positions of the devices are unknown. The approach leverages the concepts used for training Kohonen self-organizing feature maps (SOMs) to generate virtual coordinates for geographic routing. In our evaluation, the approach achieves delivery success rates of over 90% in 10% of the configurations studied, and a specific configuration achieved delivery success rates over 95% in 90% of its individual simulations. These are very promising results in relation to our baseline of 96.5% success when device locations are known perfectly. The approach also benefits from SOMs' resilience to changes in network topology, their capacity for distributed training, and from the potential to significantly reduce training overhead by refining the training strategy.

The two main areas for future work on the approach are reducing the communication overhead during self-organization, and distributing the training process. We have suggested techniques for both areas as well as arguments for why the SOMA approach should perform well in more complex future evaluations. Additional simulations in a more extensive simulation framework will verify these claims. General characteristics of self-organizing maps and their specific relationships with the task at hand, however, indicate that the SOMA virtual coordinates approach can be developed to provide distributed, adaptable, and efficient geodesic routing for networks of mobile nodes that do not know their true physical locations.

REFERENCES

[1] J. Blumenthal, M. Handy, F. Golatowski, M. Haase, and D. Timmerman, "Wireless sensor networks: New challenges in software engineering", *ETFA*, 2003.

[2] XSOM/WSOM program documentation, C. Borgelt, http://fuzzy.cs.uni-magdeburg.de/~borgelt/doc/somd/somd.html, christian.borgelt@cs.uni-magdeburg.de, Neural Networks and Fuzzy Systems Working Group, Otto-von-Guericke-University of Magdeburg.

[3] S. Capkun, M. Hamdi, and J. P. Hubaux, "GPS-free positioning in mobile ad-hoc networks", *Proc. Hawaii International Conference on System Sciences*, Jan. 2001.

[4] B. Deb, S. Bhatnagar, and B. Nath, "A topology discovery algorithm for sensor networks with applications to network management", *Proc. IEEE CAS Workshop on Wireless Communication and Networking*, Pasadena, CA, Sep. 2002.

[5] Z. Haas, "A new routing protocol for the reconfigurable wireless networks", *Proc. IEEE Int. Conference on Universal Personal Communications*, Oct. 1997.

[6] S. Haykin, Neural Networks: A Comprehensive Foundation, Second Edition, Prentice Hall, New Jersey, 1999.

[7] X. Hong, K. Xu, and M. Gerla, "Scalable routing protocols for mobile ad hoc networks", *IEEE Network Magazine*, Vol. 16, No. 4, July/August 2002.

[8] C. Intanagonwiwat, R. Govindan, and D. Estrin "Directed diffusion: A scalable and robust communication paradigm for sensor networks", *Mobile Computing and Networking*, pp. 56-67, 2000.

[9] F. Kuhn, R. Wattenhoffer, Y. Zhang, and A. Zollinger, "Geometric ad-hoc routing: Of theory and practice", *Proc. 22nd ACM Symposium on Principles of Distributed Computing (PODC)*, 2003.

[10] S. Kumar and R. Miikkulainen, "Confidence based dual reinforcement q-routing: An adaptive online network routing algorithm", *Proc. 16th International Joint Conference on Artificial Intelligence (ICJAI-99)* (Stockholm, Sweden, 1999) Kaufmann, San Francisco, CA, pp. 758-763, 1999.

[11] N. Linial, E. London, and Y. Rabinovich, "The geometry of graphs and some of its algorithmic applications", *Combinatorica*, Vol. 15, No. 2, pp. 215-245, 1995.

[12] T. Moscibroda, R. O'Dell, M. Wattenhofer, and R. Wattenhofer, "Virtual coordinates for ad hoc and sensor networks", *Proc. 2nd Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, 2004.

[13] R. Nagpal, "Organizing a global coordinate system from location information on an amorphous computer", Tech. Report AI Memo No. 1666, MIT Artificial Intelligence Laboratory, 1999.

[14] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers, *Computer Communications*, pp. 234-244, October 1994.

[15] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic routing without location information", *Proc. ACM MOBICOM*, Sep. 14-19, San Diego, CA, pp. 96-108, 2003.

[16] E. Royer and C.- K. Toh, "A review of current routing protocols for ad hoc mobile wireless networks", *IEEE Personal Communications*, p. 46-55, April 1999.

[17] Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz, "Localization from mere connectivity", *Proc. MobiHoc 2003*, Annapolis, MD, June 2003.

[18] Iterative relaxation layout algorithm applied to an irregular topology. Available: http://www.touchgraph.com/TGGoogleBrowser.html