Slime Mold Inspired Path Formation Protocol for Wireless Sensor Networks*

Ke Li[1], Kyle Thomas[2], Claudio Torres[3], Louis Rossi[3], and Chien-Chung Shen[1]

[1] Computer & Info. Sciences
[2] Chemical Engineering
[3] Mathematical Sciences
University of Delaware, Newark, DE, USA
{kli,cshen}@cis.udel.edu, kcthomas@udel.edu, {torres,rossi}@math.udel.edu

Abstract. Many biological systems are composed of unreliable components which self-organize efficiently into systems that can tackle complex problems. One such example is the true slime mold *Physarum polycephalum* which is an amoeba-like organism that seeks food sources and efficiently distributes nutrients throughout its cell body. The distribution of nutrients is accomplished by a self-assembled resource distribution network of small tubes with varying diameter which can evolve with changing environmental conditions without any global control. In this paper, we use a phenomenological model for the tube evolution in slime mold and map it to a path formation protocol for wireless sensor networks. By selecting certain evolution parameters in the protocol, the network may evolve toward single paths connecting data sources to a data sink. In other parameter regimes, the protocol may evolve toward multiple redundant paths. We present detailed analysis of a small model network. A thorough understanding of the simple network leads to design insights into appropriate parameter selection. We also validate the design via simulation of large-scale realistic wireless sensor networks using the QualNet network simulator.

1 Introduction

One fundamental design issue in wireless sensor networks (WSNs) is, given an arbitrary deployment of sensor nodes, how to connect the source nodes to the sinks so that data collected by the source nodes flow efficiently and reliably to the sinks. To facilitate such data transfer, a path formation protocol becomes essential that will determine multi-hop routes between source nodes and sink nodes. We avoid strategies that rely upon global coordination and optimization because such approaches will not scale well in large networks or networks where nodes are added or deleted dynamically, or where connections form or break easily. To seek sound solutions that rely upon local interactions, we borrow mechanisms from similar problems in the natural world and adapt them to suit the challenges of sensor networks. The natural world is full of resource distribution networks

* Preliminary work of the paper appeared as a 2-page poster in the 3rd IEEE Int. Conf. on Self-Adaptive and Self-Organizing Systems (SASO 2009).

M. Dorigo et al. (Eds.): ANTS 2010, LNCS 6234, pp. 299–311, 2010.

such as circulatory, respiratory and nervous systems in animals that are elegant solutions to very specific and complex problems. We draw our inspiration from true slime mold *Physarum polycephalum*, a biological system that has properties more aligned with the requirements of sensor networks. In particular, the dynamics of *Physarum polycephalum* constructs resource distribution networks in response to environmental conditions. This is distinct from circulatory systems which are hard-wired for a particular physiology.

Slime mold *Physarum polycephalum* is a multinuclear, single-celled organism that has properties making it ideal for the study of resource distribution networks [1]. While the organism is a single cell, slime mold can grow to tens of centimeters in size so that it can be studied and manipulated with modest laboratory facilities. The presence of nutrients in the cell body triggers a sequence of chemical reactions leading to oscillations along the cell body [5]. Tubes self-assemble which are perpendicular to the oscillatory waves to create networks linking nutrient sources throughout the cell body. There are two key mechanisms in the slime mold life cycle that transfer readily to WSNs. First, during the growth cycle, the slime mold will explore its immediate surroundings with pseudopodia via chemotaxis to discover new food sources. We have applied this mechanism successfully to WSNs in a previous work treating data sources and sinks as singular potentials [2]. In this current work, we exploit the second key mechanism which is the temporal evolution of existing routes through nonlinear feedback to efficiently distribute nutrients throughout the organism. In slime mold, it can be shown experimentally that the diameters of tubes carrying large fluxes of nutrients grow to expand their capacity, and tubes that are not used decline and can disappear entirely.In the context of WSNs, the diameter of a tube may be used to denote the Quality of Service (QoS) of a wireless channel.

To evaluate the merits of path formation, we focus on three metrics that directly measure the network performance: expected hop count, fault tolerance, and node degree distribution. The *expected hop count* denotes the efficiency of data flow in a network by measuring the expected number of nodes a data packet must traverse to reach the sink. We compute the expectation because the protocol may construct topologies with multiple disjoint routes. The *fault tolerance* is a gross indicator of network robustness measured by the probability that a node failure will disconnect at least one source from the network. Finally, we measure the *node degree distribution* for the network as an indicator of connectivity.

This paper is structured as follows. In Section 2, we describe the model of Tero et al. for solutions of *Physarum polycephalum* to mazes and map it to the path formation problems of WSNs. In Section 3, we describe the path formation protocol based on the slime mold model. In Section 4, we verify our model by independently computing all possible stationary states and their stability on a very small network. In Section 5, we perform large scale simulations and measure the protocol performance with quantitative metrics of expected hop count, fault tolerance, and node degree distribution as functions of network parameters.

2 A Phenomenological Model for Slime Mold

We base our path formation protocol for WSNs on the phenomenological model proposed by Tero et al. for tube evolution in *Physarum polycephalum* to reproduce the slime mold maze solving experiments [3,6]. In recent article on applications of their slime-inspired algorithm, Tero et al. independently acknowledge the suitability of this type of model for sensor networks [7]. The model captures the evolution of tube capacities in an existing network through a dynamical system as follows. The original model consists of a set of nodes representing either a food source or a junction in the maze connected by a set of edges. Along each edge, the flow of nutrients from node i to node j is represented as q_{ij}, which is directional and $q_{ij} = -q_{ji}$. Flow through the network is driven by a pressure at each node p_i. The cross-sectional size of each tube is represented by D_{ij} ($D_{ij} = D_{ji}$). At each node, a Kirchhoff law is required to enforce conservation of nutrient flux at each junction. All above variables evolve in time as follows:

$$q_{ij} = \frac{D_{ij}}{L_{ij}}(p_i - p_j), \tag{1a}$$

$$\sum_{j \in N_i} q_{ij} = m_i, \tag{1b}$$

$$\frac{dD_{ij}}{dt} = f(|q_{ij}|) - rD_{ij}, \tag{1c}$$

where L_{ij} is the length of the tube, N_i represents the set of neighbors for node i, m_i is the amount of nutrient flux into the network from i, $f(x)$ is either an exponential or sigmoidal function describing the rate of growth of the tube size in response to nutrient flow rate, and r is a tube size linear decay rate. The system (1) contains a continuity equation (Kirchhoff Law) (1b) balancing the flow out of a node with the source value. In Tero et al. [6], nutrient sources were designated to be $m_i = 1$ at the entrance or $m_i = -1$ at the exit to drive flow through a maze. At junctions within the maze, $m_i = 0$.

We use the Tero model as a foundation for the path formation protocol for WSNs. While the distinction of identical nutrient sources as either $m_i = +1$ or $m_i = -1$ in the slime mold seems arbitrary, this aspect of algorithm maps well to WSNs which have sensors that are data sources and sinks that represent highly capable nodes collecting source data. The set N_i of neighbors denotes the set of nodes within the transmission radius of node i. Thus, there is a communication link between node i and all members of N_i. In the proposed protocol, we assume symmetric links with all identical transmission radii. While the tube length L_{ij} could be a useful tuning parameter mappable to special restrictions on network connections, we do not pursue these issues in this paper and simply assume equal L_{ij}'s throughout the network. We represent $f(x)$ as sigmoid function slightly differently from Tero et al. as

$$f(x) = rD_{\max}\frac{a|x|^{\mu}}{1 + a|x|^{\mu}}, \tag{2}$$

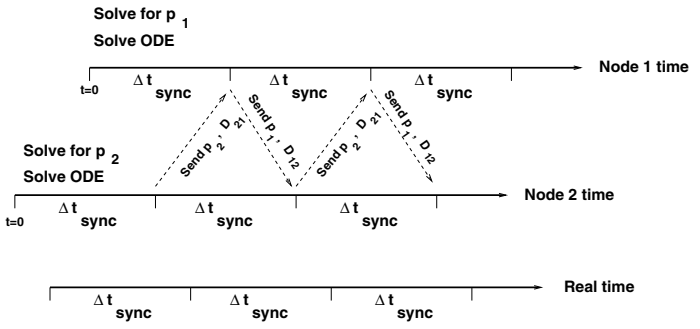

Fig. 1. Schematic diagram of local computation followed by synchronization of protocol variables. Nodes 1 and 2 perform computations independently within a fixed cycle of Δt_{sync}, and share information afterward.

where $D_{\max}$ specifies the upper limit on D_{ij}'s. The exponent μ plays a critical role in the behavior of this model, which will be discussed later. The parameter a describes the shape of the sigmoid and has little impact on network performance. Without loss of generality, we simply select all L_{ij}'s, r, D_{max}, and a to be 1 when calculating numerical solutions to the model as well as simulating the path formation protocol.

Like ant colony optimization (ACO), the slime mold model includes dynamic positive and negative feedback mechanisms (1c). The key distinction is that the model permits nearly simultaneous coordination through flow continuity (1a,1b), which provides both the advantage that more information is available, and the disadvantage that the pressure equation must be solved iteratively as will be discussed in Section 3.2. The solution to this system may be used to move data from sources to data sink(s) in WSNs following a multi-path routing scheme. The outward fluxes from a node i guide how packets are routed through the network. If we denote O_i to be the subset of $k \in N_i$ such that $q_{ik} > 0$, O_i then represents the next-hop neighbors of node i on a route toward the sink. We define the ratio

$$w_{ij} = \frac{q_{ij}}{\sum_{k \in O_i} q_{ik}}. \tag{3}$$

If node i has C data packets to send, Cw_{ij} packets (rounding as necessary) would be sent to node j for each $j \in O_i$. While the w_{ij}'s do not necessarily represent probabilities, we could pose a stochastic routing protocol by routing each packet arriving at node i to node j with probability w_{ij}.

3 Protocol Description

Although model (1) is globally coupled via the pressure field, the inspired path formation protocol solves the model in a distributed manner. Each node i maintains its local states of own pressure p_i, perceived neighbor pressures p_j's, and

tube size D_{ij}'s for all $j \in N_i$. We initialize pressures uniformly as 1.0 (except for the sink as 0), with D_{ij}'s randomly selected between 0 and $D_{\max}$. Each node then performs local computations in a fixed cycle of Δt_{sync}, solving model (1b,1c) for pressure and link diameters. At the end of each Δt_{sync} cycle, a node broadcasts its latest computing results to neighbors, which allows globally coupled information to propagate across the network. Meanwhile, each node keeps updating perceived neighbor pressures to be used in future local computation. The loop of local computation followed by states synchronization at fixed interval of Δt_{sync} is demonstrated in Figure 1. Notice that the localized path formation protocol solves model (1) cooperatively among neighboring nodes and in real time without explicit clock synchronization across the network. All that is required is that clocks run forward at approximately the same rate in every node.

3.1 Local Data Structures

Each node i locally maintains a node type ($nodeType_i$), a net flux ($netFlux_i$), a pressure value (p_i), and a *neighbor* table ($nTab_i$). The value of $nodeType_i$ could be *SINK*, *SOURCE*, or *OTHER* (as potential relays). The $netFlux_i$ specifies the pure flux m_i flowing out of node i into the network, which is positive at any data source, negative at the data sink, and zero at any other node. The pressure value of each node other than the sink starts with 1.0 and evolves based on model (1), while the sink's pressure remains 0 at all time. The *neighbor* table entry $nTab_i(j)$ corresponds to the 1-hop neighbor j of node i, with the form of $\langle p_j, D_{ij}, q_{ij}, L_{ij} \rangle$. The *neighbor* table keeps track of local dynamics of the evolving system and thus forms the basis for solving model (1).

3.2 Local Computation

Between synchronizations of pressures and link diameters with neighbors, each node i performs local computation in R rounds to solve model (1) for pressure p_i, neighbor link flux q_{ij}'s, and diameter D_{ij}'s, based on the following mathematical methods.

To solve for p_i locally, we substitute the flow model (1a) into the pressure equation (1b):

$$p_i \leftarrow \frac{m_i + \sum_{j \in N_i} \frac{D_{ij}}{L_{ij}} p_j}{\sum_{j \in N_i} \frac{D_{ij}}{L_{ij}}}. \tag{4}$$

Similar to the Jacobi algorithm for solving diagonally dominant linear systems, the above method of solving Kirchhoff law (1b) in terms of pressure is *translation invariant*, meaning that one can take a pressure solution and add the same constant to all pressures to obtain another solution. Therefore, we fix the pressure to be 0 at the sink node which does not satisfy Kirchhoff's law. Conservation of flux is already imposed and fixing the pressure at the sink node guarantees a unique pressure solution. With (1b) satisfied at all nodes except for the sink, we

then numerically solve the evolution equation for D_{ij} iteratively using the first order implicit scheme proposed by Tero:

$$D_{ij}^{n+1} = \frac{D_{ij}^{n} + \Delta t f(|q_{ij}^{n}|)}{1 + \Delta t r}, \tag{5}$$

where the fluxes q_{ij} are determined from the solved pressures p_i, Δt is the duration of one discrete time step (or iteration), and D_{ij}^{n} is the value of D_{ij} at the n^{th} iteration.

Each round of the local computation works as follows. First, every node i except for the sink calculates its pressure p_i by solving the linear equation (Kirchhoff's law), given neighbors' p_j's and D_{ij}'s based on (4), while the sink remains zero pressure as the minimum pressure level. Next, for each of its neighbor j, node i determines the corresponding flux q_{ij} with the freshly calculated p_i. Then it solves the adaptive system (1c) for D_{ij} based on (5) in S iterations, with each iteration's result based on the previous one and fed into the next. Since each iteration signifies Δt time, the total time for R rounds of S iterations is thus $T = R \times S \times \Delta t$ before the next synchronization. We require that $\Delta t_{\text{sync}} \geq T$.

3.3 Synchronization

Before each cycle of local computation, every node i shares its pressure p_i and synchronizes D_{ij}'s with neighbors by broadcasting a one-hop synchronization (SYNC) packet. The SYNC packet contains the sender's *id* and current pressure, together with the latest *neighbor-diameter* table, including the total number of entries (*#entries*) followed by each entry $\langle n_j, D_{id,n_j} \rangle$. In particular, D_{id,n_j} represents the diameter of link to its neighbor n_j, retrieved from the sender's *neighbor* table. The initial synchronization also serves as a hello packet for neighbor discovery, whose *neighbor-diameter* table is empty with $\#entries == 0$.

Upon receiving a SYNC packet, each node tries to synchronize its *neighbor* table in terms of neighbor pressure and link diameter based on information from the packet. In case of the first time to hear from sender j, node i will insert in its *neighbor* table a new entry for j. Specifically, D_{ij} in the new entry is either assigned as the corresponding D_{ji} from the packet's *neighbor-diameter* table if it exists, or otherwise initialized as a random value $rand(i + j)$ within $[0, D_{max}]$. Notice that the latter case applies when receiving the initial batch of SYNC packets which also serve for neighbor discovery, and the random diameter uniquely dependent on $(i+j)$ ensures that $D_{ij} = D_{ji}$ at the start. In case sender j already exists in node i's *neighbor* table, p_j is updated and D_{ij} synchronized as the mean of itself and D_{ji} from the packet.

3.4 Issue of Asymmetric Neighborhood

Due to unreliability of wireless communications, synchronization packets may get lost or corrupted. This is annoying when lost packets assist in neighbor discovery, which could cause "asymmetric neighborhood", *e.g.*, node i can hear (sometimes) from node j thus recording j in its *neighbor* table and counting p_j

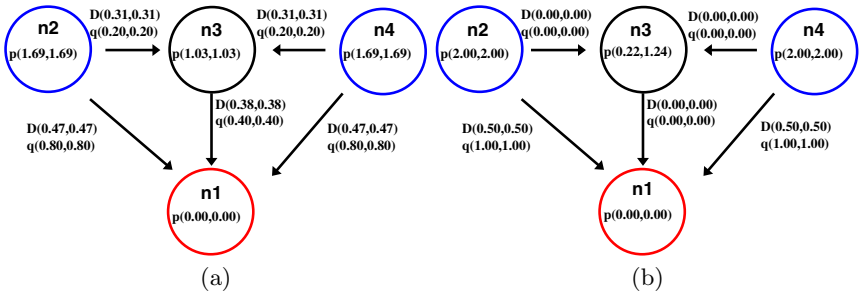

Fig. 2. Comparison between steady states from numerical solutions (left value) and QualNet simulations (right value) on the 4-node network under different μ values. (a) $\mu = 0.5$, (b) $\mu = 2$.

and D_{ij} during its local computations, while node j could not receive successfully from i thus never taking i as its neighbor. This problem is aggravated when node i decides through local computation to send majority of its outgoing flux to its seeming neighbor j, while node j is ignorant of the large flux from node i, which could potentially disrupt the global flow continuity (1b) and render the protocol unable to converge. Asymmetric neighborhood becomes widespread as the number of nodes within transmission range increases which causes severer channel contentions.

Although packet loss may be mitigated via MAC layer schemes like random jitters, it could never be eliminated as long as the channel is unreliable. We take an alternative approach to contain the bad effect of packet loss so as to prevent the asymmetric neighborhood, by introducing the neighborhood threshold (*neighThresh*). Every node tracks the number of SYNC packets sent out (*#pktSent*), as well as received from each neighbor (*#pktRcvdt*) recorded within each neighbor entry of its *neighbor* table. During local computation, nodes only consider "true" neighbors that satisfy:

$$\#pktRcvd > \#pktSent \times neighThresh \tag{6}$$

where $neighThresh \in [0, 1]$, and needs to be carefully selected. In practice, the neighborhood threshold is usually chosen within $[0.5, 1]$ — The more contentions or worse channel quality, the bigger the threshold used. The basic rationale of this approach is the assumption that the channel quality between two nodes is roughly symmetric when both nodes transmit using the same power within a short time period.

4 Validation with Stability Analysis

4.1 Validation

To validate the path formation protocol, we systematically studied the stationary solutions of the dynamical system (1) on a 4-node small network, and compared

exact numerical solutions with results from distributed calculations in realistic QualNet [4] simulations. Details on calculating exact solutions are discussed in Section 4.2.

The 4-node network consists of one sink n_1, two sources n_2 and n_4 each with 1 unit of flux to be sent to the sink, and one potential relay n_3, with network layout shown in Figure 2. By numerically solving the model, we found 9 steady states for $\mu = 0.5$ (demonstrated in Figure 3 inside ovals), and 6 for $\mu = 2$.We ran QualNet simulations under same set of parameters with different random seeds. For $\mu = 0.5$, all trials converged to the same single steady state shown in Figure 2(a), the reason of which will be discussed in Section 4.2. For $\mu = 2$, different steady states were found by simulations. Figure 2 compares one corresponding steady state between the solution of numerical method and QualNet simulation for each μ value. We find that the exact numerical solutions and the realistic QualNet simulation results agree on D_{ij}'s and q_{ij}'s. So do they on almost all pressure p values except for node n_3 when $\mu = 2$, in which state both sources n_2 and n_4 send all fluxes directly to the sink n_1 without passing through n_3. Therefore n_3 has the "freedom" of choosing any positive value as pressure, since it will not be used for relay. This "freedom" doesn't impact the protocol because it only happens when the flux through the node is zero. From the mathematical point of view, this case corresponds to the existence of a nontrivial, homogeneous solution when we solve (1a) and (1b) for the pressure.

4.2 Linear Stability Analysis

To understand the stability of solutions to model (1), we analyze solutions to the nonlinear system of equation $\mathcal{F}(D_{ij}) = f(|q_{ij}|) - rD_{ij} = 0$ when setting $\frac{dD_{ij}}{dt} = 0$ in (1). The stationary solution to $\mathcal{F}(D_{ij}) = 0$ is defined as an implicit nonlinear system, where the implicit part (q_{ij}) is obtained by solving a nested linear system for pressure values using system (1). After calculating solutions to $\mathcal{F}(D_{ij}) = 0$, we then compute the Jacobian matrix $\left[\frac{\partial \mathcal{F}_{ij}}{\partial D_{kl}}\right]$ evaluated at the stationary point of interest. The eigenvalues and eigenvectors of each solution determine the stability of the system. If all eigenvalues are negative, the

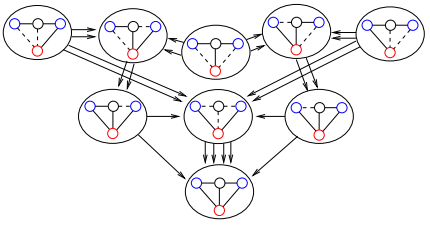

Fig. 3. Stability diagram for the 4-node network with $\mu = 0.5$. The topology of each of these 9 steady solutions is shown inside the oval, with solid lines indicating links with $D_{ij} \neq 0$ and dotted lines for links with $D_{ij} = 0$. Each pair of arrows represents the growth of a small positive and negative amplitude perturbation along an eigenvector.

solution is a stable steady state. Otherwise when there are any positive eigenvalues, the solution is unstable, which will evolve to a different steady state, with the linear evolution described by the eigenvector corresponding to the positive eigenvalue.

The complete understanding of small problems gives us useful insights into how parameter values affect the behavior of the whole network. In particular, the parameter μ dominates the evolution of links in the network. Generally speaking, for $\mu = 2$ (in general $\mu > 1$) all single route solutions with positive D_{ij} values are stable. On the other hand, for $\mu = 0.5$ (in general $\mu < 1$) only the solution where all links are used is stable, which corresponds to the single steady state reached by QualNet simulations. Since there are both stable and unstable numerical solutions when $\mu = 0.5$, we also built the stability diagram over all 9 solutions shown in Figure 3, where the perturbations were computed using the eigenvectors related to the positives eigenvalues. We find that for $\mu = 0.5$ the initialization does not affect the final stable state at all, which should always be the all-link solution illustrated by Figure 2(a). This mathematical analysis also explains why all QualNet simulations converged to this stable solution when $\mu = 0.5$. Finally, these general properties transfer to larger networks.

5 QualNet Simulation

We performed QualNet simulations on 100 and 300-node networks with one sink and varying numbers of sources. All nodes were random distributed over a terrain of size 1000×1000 m^2. Each node in the network was equipped with a radio transceiver capable of transmitting signals up to approximately 80 meters. The underlying wireless channel had a data rate of 2 Mbps, using the two-ray path loss model without fading. IEEE 802.11 DCF was used as the MAC layer protocol, and IP as the network layer on top of which our protocol ran. No application traffic was employed since the sole purpose was to connect source nodes to the data sink. The neighborhood threshold is 0.8 to avoid neighborhood asymmetry. For each cycle of local computation, the number of round $R = 10$, iteration $S = 1$, and $\Delta t_{\text{sync}} = 0.01$ sec. The following three quantitative metrics are designed to measure qualities of resulting network connectivity under $\mu = 0.5$ and $\mu = 2$ with different portions of sources:

- **Expected hop count:** indicates the *efficiency* of the connectivity in terms of hop count distance between a source and the sink. The smaller the hop count, the higher the efficiency. Let $W = [w_{ij}]^T$ be the Markov transition matrix with w_{ij} defined in Equation (3). Let $\boldsymbol{e}_l$ be a unit vector consisting of all zeros except at element l. Then $W\boldsymbol{e}_l$ is a vector whose j^{th} element denotes the probability of a data packet starting from node l and arriving at node j in one hop. Similarly, $W^k\boldsymbol{e}_l$ is a vector of probabilities for k hops. Therefore, the expected hop count from source node l to sink node s is

$$E(l, s) = \sum_{k=1}^{\infty} kW^k\boldsymbol{e}_l. \tag{7}$$

Table 1. Fault tolerance of a 300-node network with 1 sink and varying numbers of sources

source	10%		20%		30%		40%		50%	
μ	0.5	2	0.5	2	0.5	2	0.5	2	0.5	2
1-fault	1	.9851	.9916	.9874	.9856	.9761	.9832	.9609	.9799	.9530
2-fault	.9998	.9700	.9826	.9739	.9709	.9616	.9659	.9222	.9595	.9070

- **Fault tolerance:** indicates the *robustness* of the connectivity. We define the x-fault tolerance to be the probability that removing x relay nodes will *not* yet result in any source being disconnected from the sink. The higher the probability, the better the robustness.
- **Degree distribution:** defines $P(d)$ as the probability of a node in the network having a degree of d. The presence of a small number of high degree nodes may indicate a greater dependence on a few "gateway nodes" for network connectivity.

Figure 4 shows snapshots of steady state connectivity under $\mu = 0.5$ and $\mu = 2$ over a 100-node randomly generated network with one data sink and three data sources. Figure 5 depicts a 300-node network with one data sink and thirty data sources. The width and color of a link signify the value of corresponding flow q_{ij}. The arrow of the link shows the direction of the flow. We find that the network with $\mu < 1$ (e.g., $\mu = 0.5$) evolves towards multi-route robust connectivity, whereas the network with $\mu > 1$ (e.g., $\mu = 2$) evolves into single-route efficient topology. This verifies that the exponent μ is a critical parameter

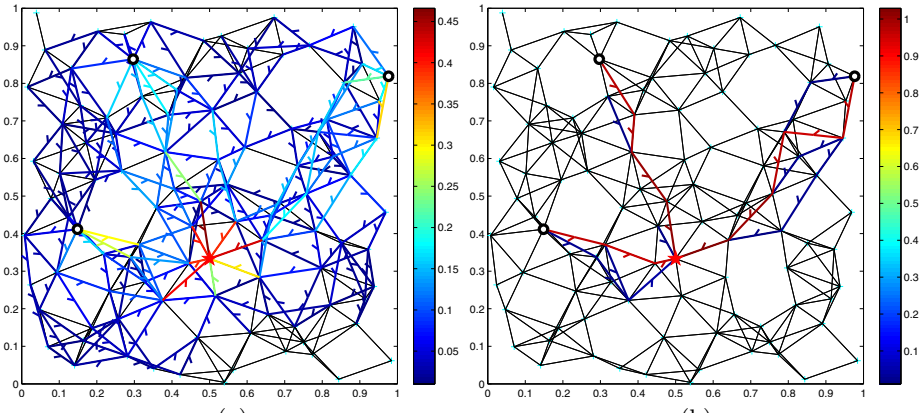

Fig. 4. Steady states obtained by QualNet for a 100-node network, including 1 sink (the star near the center) and 3 sources (the bold circles) with link color and width signifying the flux value under (a) $\mu = 0.5$ and (b) $\mu = 2$

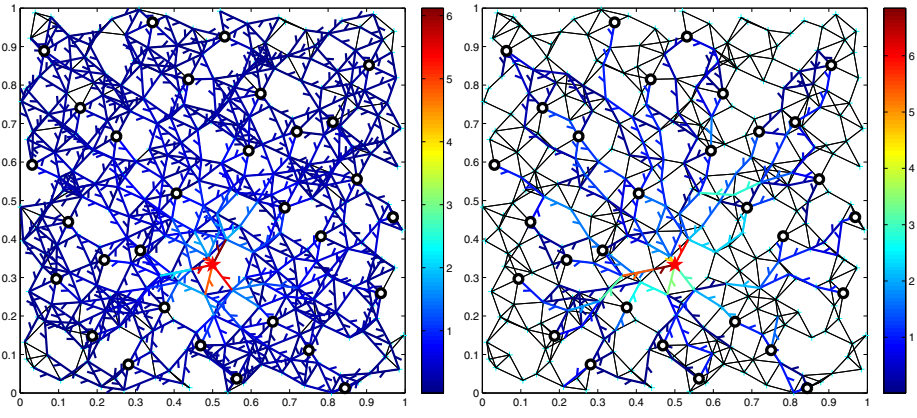

Fig. 5. Steady states obtained by QualNet for a 300-node network, including 1 sink (the star near the center) and 30 sources under (a) $\mu = 0.5$ and (b) $\mu = 2$

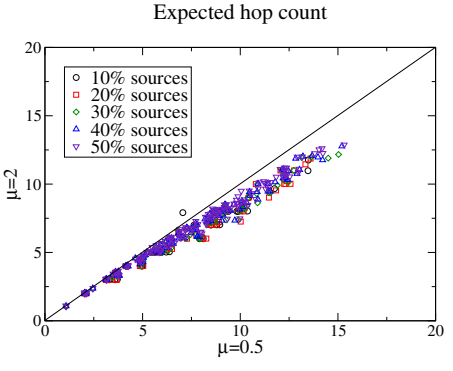

Fig. 6. Expected hop count for a 300-node network with 1 sink and 10% to 50% sources. Each point corresponds to two expected number of hops between one of the sources and the sink, with x-coordinate for $\mu = 0.5$ and y-coordinate for $\mu = 2$.

for balancing efficiency and robustness by determining whether a single route or multiple routes are created between sources and the sink.

Figure 6 reflects the efficiency of connectivity for a 300-node network with one sink and 10% to 50% of sources, with each data point signifying two expected hop counts between one source and the sink defined in (7) for $\mu = 0.5$ (x-coordinate) and $\mu = 2$ (y-coordinate). Although all points spread close to the diagonal, many lie below the diagonal towards the x-axis of $\mu = 0.5$, implying that the robustness provided by $\mu = 0.5$ comes at the cost of increased hop counts. Linear regression of the data shows that $\mu = 0.5$ requires approximately

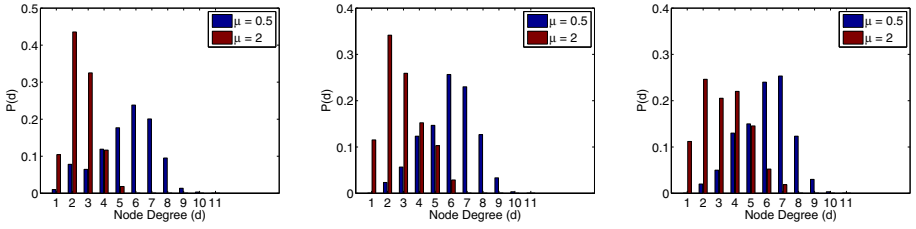

Fig. 7. Distribution of node degree for (a) 10%, (b) 30%, and (c) 50% sources on a 300-node network with 1 sink

10% more hops than $\mu = 2$. Table 1 presents the 1 and 2-fault tolerances for the same networks, showing that $\mu = 0.5$ achieves better robustness against node failure than $\mu = 2$, and that the network becomes less fault tolerant with more sources added. Figure 7 presents the distribution of node degree. As we would expect, $\mu = 0.5$ leads to a higher proportion of high degree nodes in a multi-route connectivity, whereas $\mu = 2$ leads to a greater proportion of low degree nodes in a single-route topology. The distinction blurs as more sources join the network.

6 Conclusion

In this paper, we design a slime mold inspired, self-organizing path formation protocol for WSNs. The localized protocol is effective in iteratively solving the underlying mathematical model which is globally coupled through the pressure field. The critical parameter μ controls whether single ($\mu > 1$) or multi-route ($\mu < 1$) connections emerge. We validate the protocol through linear stability analysis on a small model network, as well as via simulations on large realistic WSNs with one sink and varying numbers of sources. Future work will focus on extending the protocol to deal with real network traffic, exploring L_{ij} as a measure of link quality, and modifying the biologically inspired model to address WSNs with coordinated routing to multiple sinks.

Acknowledgements. This work is supported in part by National Science Foundation under grants CCF-0726556 and CNS-0347460.

References

1. Ben-Jacob, E., Cohen, I.: Cooperative organization of bacterial colonies: From genotype to morphotype. Annual Review of Microbiology 52, 779–806 (1998)
2. Li, K., Thomas, K., Rossi, L.F., Shen, C.C.: Slime-mold inspired protocol for wireless sensor networks. In: Proc. of the 2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO), pp. 319–328. IEEE Press, Los Alamitos (2008)

3. Nakagaki, T., Yamada, H., Toth, A.: Maze-solving by an amoeboid organism. Nature 407(6803), 470–470 (2000)
4. Scalable Network Technologies, Inc.: QualNet Simulator, http://www.scalable-networks.com
5. Stewart, P.A.: The organization of movement in slime mold plasmodia. In: Primitive Motile Systems in Cell Biology, pp. 69–78. Academic Press, London (1964)
6. Tero, A., Kobayashi, R., Nakagaki, T.: A mathematical model for adaptive transport network in path finding by true slime mold. J. Theor. Biol. 244, 553–564 (2007)
7. Tero, A., Takagi, S., Saigusa, T., Ito, K., Bebber, B.P., Fricker, M.D., Yumiki, K., Kobayashi, R., Nakagaki, T.: Rules for biologically inspired adaptive network design. Science 327, 439–442 (2010)