A Formal Performance Modeling Framework for Bio-inspired Ad Hoc Routing Protocols

Muhammad Saleem
Center for Advanced Studies in Engineering
Islamabad 44000, Pakistan
msaleem@case.edu.pk

Syed Ali Khayam
WisNeT
SEECS-NUST
Rawalpindi, Pakistan
khayam@niit.edu.pk

Muddassar Farooq
nexGIN RC
NUCES-FAST
Islamabad 44000, Pakistan
muddassar.farooq@nu.edu.pk

ABSTRACT

Bio-inspired ad hoc routing is an active area of research. The designers of these algorithms predominantly evaluate the performance of their protocols with the help of simulation studies. Such studies are mostly scenario and simulator specific and their results cannot be generalized to other scenarios and simulators. Therefore, we argue that mathematical tools should be utilized to develop a consistent, provable and compatible formal framework in order to provide an unbiased evaluation of Bio-inspired ad hoc routing protocols. Motivated by this requirement, in this paper, we develop a probabilistic performance evaluation framework that can be used to model the following key performance metrics of an ad hoc routing algorithm: (1) routing overhead, (2) route optimality, and (3) energy consumption. We utilize this framework to model a well known Bee-inspired routing protocol for ad hoc sensor networks, BeeSensor. We also show that the proposed framework is generic enough and can easily be adapted to even model a classical routing protocol, Ad Hoc on Demand Distance Vector (AODV). The modeled metrics of the two algorithms not only allow unbiased performance comparison but also provide interesting insights into the parameters governing the behavior of these routing protocols.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: [Distributed networks, Wireless communication, Network communications, Network topology]; C.2.2 [**Network Protocols**]: [Routing protocols]; C.4 [**Performance of Systems**]: [Modeling techniques]; G.3 [**Probability and Statistics**]: [Distribution functions, Probabilistic algorithms (including Monte Carlo), Stochastic processes]

General Terms

Algorithms, Design, Performance, Theory

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.

Keywords

Mathematical Models, Routing and Layout, Swarm Intelligence, Telecommunications, Wireless Ad Hoc Networks

1. INTRODUCTION

Wireless ad hoc sensor networks is an active area of research due to their potential utilization in a large set of applications that include target field imaging, intrusion detection, weather monitoring, security and tactical surveillance and disaster management [8]. Design and development of Bio-inspired routing protocols for sensor networks has received little attention. In [3] and [15], the authors reported the ant based routing algorithms for wireless sensor networks. Recently proposed BeeSensor [12] protocol tries to combine the efficient performance of BeeHive [5, 13] with the energy efficiency of BeeAdHoc [4]. These relevant protocols, though inspired from totally different colony systems, have one important thing in common:*the designers of the protocols have predominantly evaluated the performance metrics of their protocols with the help of simulation studies only.* But a statistical survey by Kurkowski *et al.* [6] clearly identifies the serious shortcomings of simulation-based performance evaluation. Therefore, we argue that simulation studies must be complemented with mathematical evaluation for a consistent, provable, compatible and unbiased evaluation of Bio-inspired ad hoc routing protocols.

Designing a formal framework that models the key performance parameters of Bio-inspired ad hoc routing protocols is a difficult task due to a number of challenges: (1) stochastic nature of the physical medium in wireless networks, (2) continuously changing network topology, (3) random geographical placement of the sensors nodes in real networks, (4) stochastic re-broadcasting principle utilized by many ad hoc routing protocols, (5) stochastic routing mechanism employed by the Bio-inspired protocols, and last but not least (6) no agreed definition about the notion of optimality. The real daunting task is *to model the behavior of a stochastic routing protocol deployed in a stochastic environment.* In this paper we report, to the best of our knowledge, first ever formal performance evaluation framework that can be used to model the following widely-accepted performance metrics of wireless ad hoc routing algorithms [2]: (1) routing overhead, (2) route optimality, and (3) energy consumption.

The derived metrics can be parameterized in a generic model, which can then be easily adapted to a number of specific ad hoc routing protocols. In this paper, we model BeeSensor and the de facto AODV protocol [7] by utilizing our proposed framework. This effort provides us with intriguing insights

into the behavior of these protocols that could not be inferred through simulation-based evaluations. For instance, one important finding of our study is that *a purely stochastic flooding approach is inappropriate for ad hoc routing because such an approach cannot ensure route establishment even if the probability of rebroadcasting route requests is quite high.* We will discuss other insights once we introduce our model.

Related Work. As mentioned before, virtually no attempts are made to formally model the behavior of Bio-inspired routing protocols with the exception of the work of Roth [9] [10] and Zahid *et al.* [14]. The later work, however, is limited to the fixed networks only. In [10], Roth provided an analytical justification of the three pheromone update mechanisms used in Termite [11]. He has also developed an analytical framework based on the Markov chains for the analysis of probabilistic routing protocols [9].

Organization of Paper. The rest of the paper is organized as follows. Section 2 contains system description and modeling assumptions. Section 3 describes the modeling of routing overhead followed by the route optimality model in Section 4. The expressions for the total energy consumed during the transmission and reception of the packets are derived in Section 5. Finally we conclude our paper with an outlook to our future research.

2. SYSTEM DESCRIPTION AND MODELING ASSUMPTIONS

2.1 Basic Graph Terminology

A typical graph is denoted by $G(V, E)$ in which V is a set of vertices in the graph and E is the set of edges. This model can be used to represent an ad hoc network in which individual nodes are the vertices of the graph connected through wireless links (edges of the graph). In this section, definitions of basic graph-theoretic terms are provided.

Node degree: Degree of a node x, $d(x)$, represents the number of nodes directly connected with x. Minimum degree of a graph G is then defined as:

$$d_{min}(G) = min\ \{d(x)\} \qquad \forall\ \mathrm{x}\ \epsilon\ \mathrm{G}$$

A similar term is the average node degree defined as:

$$d_{avg}(G) = \frac{1}{n}\sum_{x=1}^{n} d(x)$$

A node with zero degree is an isolated node.

Connected and disconnected graphs: A graph is said to be connected if there exists at least a single path between each pair of nodes in the graph [1]; otherwise, the graph is said to be disconnected. If there exists at least k mutually independent paths between each node pair in the graph, the graph is said to be k-connected. Another interesting term is edge connectivity. A graph is said to be k-edge connected if and only if k edge-disjoint paths exist between each node pair. A k-connected graph is also a k-edge connected graph but the reverse is not always true.

2.2 Network Topology

Network topology of an ad hoc network plays the pivotal role in modeling of an upper layer protocol. As nodes are connected through wireless links, their deployment pattern and adjustment of their transmission powers is critical to keep the graph connected. Fortunately, this problem has already been addressed in Bettstetter's seminal work [1]. Assuming that N nodes are randomly distributed and connected through symmetric links, Bettstetter derived an expression that, for a given node density ρ, determines the transmission range r_0 to ensure that a randomly chosen node will have exactly n_0 neighbors. Specifically, probability P that a node has exactly n_0 neighbors is given by

$$P(d = n_0) = \frac{(\rho\pi r_0^2)^{n_0}}{n_0!} \cdot e^{-\rho\pi r_0^2}. \tag{1}$$

Putting $n_0 = 0$ in (1) gives the probability that a node selected at random will be an isolated node; i.e. $P(d = 0) = e^{-\rho\pi r_0^2}$. Expected or average degree of a node is given by the following relation:

$$E(d) = d_{avg} = \rho\pi r_0^2 - 1. \tag{2}$$

To be sure with a certain probability p that the network having $n >> 1$ nodes is connected, we must set

$$r_0 \geq \sqrt{\frac{-\ln(1 - p^{\frac{1}{n}})}{\rho\pi}}. \tag{3}$$

Equation (2) and (3) can be used to create an ad hoc network with desired on-average topology characteristics. For example, if 1000 nodes ($n = 1000$) are distributed in an area of $10^6\ m^2$, transmission radius r_0 of each node must be set to a value greater than 60 m so that the network is connected (with 99 % probability) and each node has an average degree of 11.

2.3 Modeling Assumptions

We assume a dense network with an average degree of d_{avg} and having symmetric links between the nodes. The nodes are deployed according to a homogeneous Poisson distribution with the node density ρ and each node having a transmission radius r_0. For performance evaluation under ideal routing conditions, we ignore Medium Access Control (MAC) layer contention/collisions and link quality variations. Thus we study the behavior of a routing algorithm in a purely geometric graph model by ignoring the problems that are beyond the control of a network layer protocol.

3. ROUTING OVERHEAD MODEL

3.1 Generic Model

In this section, we describe and model the route discovery mechanism of typical reactive ad hoc routing protocols. In reactive protocols, route discovery is initiated by a node, called source node, when it has some data to deliver to an unknown destination. The source node broadcasts a route request (RREQ) packet to all the nodes within its transmission radius. In a pure flooding protocol, each intermediate node rebroadcasts this RREQ packet until it reaches the intended destination. However, to keep the model generic, we assume that intermediate nodes rebroadcast the RREQ with some probability p_r. Henceforth, we refer to p_r as the *rebroadcasting probability.*

We define *Routing overhead* as: *the number of RREQs that a protocol transmits in the network up to a particular number of hops, h, during the route discovery phase.* Our definition of *Routing overhead* is a generic definition that also incorporates the distinguishing feature of many ad hoc routing

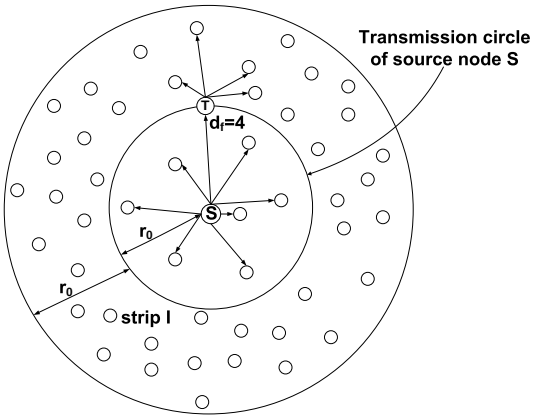

Figure 1: Route discovery in an ad hoc network

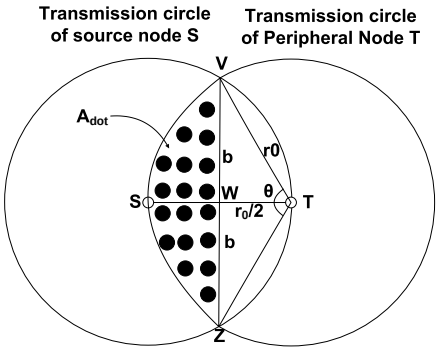

Figure 2: Overlapping transmission circles of Node S and T

protocols that do not flood the RREQs in the entire network e.g. expanding ring search algorithms.
Consider a source node S that initiates a route discovery process (see Figure 1) by broadcasting an RREQ packet to all its neighbors. In the rest of this section, we derive an expression for the expected routing overhead incurred during this process in a reactive ad hoc routing algorithm.

3.1.1 Routing Overhead in terms of Expected Forward Degree

The initial RREQ broadcasted by the source is received by d_{avg} nodes, the average degree of a node. Each one of d_{avg} neighbors rebroadcasts the RREQ with probability p_r and, hence, the first hop rebroadcasting nodes equal $p_r \times d_{avg}$. The receiving nodes rebroadcast the RREQ with probability, p_r, and the process continues. To compute total expected routing overhead, we can accumulate the total number of RREQs, C_p, injected into the network up to h hops from the source node. This cumulative term is given as:

$$C_p = 1 + p_r d_{avg} + p_r^2 d_{avg} d_f + p_r^3 d_{avg} d_f^2 + \ldots + p_r^h d_{avg} d_f^{h-1}. \quad (4)$$

Individual terms in (4) represent the number of rebroadcasting nodes at each hop. We have introduced a new term in the equation, d_f. We call it the *expected forward degree* of a node and define it as: *"the number of new neighbors of a node that will receive the RREQ of that node and rebroadcast it to the next hop with probability p_r"*. This definition is a direct consequence of the fact that every node receiving an RREQ is not likely to rebroadcast it. For example, the rebroadcast of node T (see Figure 1) is received by the nodes in strip I as well as the nodes located within the transmission circle of source node S. However, the neighbors of node S have already received a copy of RREQ and hence are bound to drop the duplicate RREQ. It is important to note here that we do not consider the case of super flooding in which intermediate nodes might decide to rebroadcast the duplicate RREQ if it is coming from a better path e.g. shortest hop path.
The initial RREQ broadcasted by the source node is received by d_{avg} nodes and each one of them in turn rebroadcasts the RREQ with probability p_r. Hence, the first hop rebroadcasting nodes equal $p_r d_{avg}$. As each one of $p_r d_{avg}$ nodes can reach d_f nodes on the average, the total number of new nodes that will receive the first hop rebroadcasts are $p_r d_{avg} \times d_f$. Now the receivers ($p_r d_{avg} \times d_f$) are likely to repeat it with probability p_r, hence the total number of rebroadcasting nodes at 2 hops from the source equals $p_r \times (p_r d_{avg} \times d_f) = p_r^2 d_{avg} d_f$ (see the third term in (4)). In this way, we calculate the number of rebroadcasting nodes at each step using the *expected forward degree* of the nodes. Getting back to (4), we have a total of $h + 1$ terms. The closed form of the expression is

$$C_p = 1 + p_r d_{avg} \sum_{i=0}^{h-1} (p_r d_f)^i. \quad (5)$$

C_p includes the broadcasting of nodes at a distance of h hops and all the previous hops' broadcasts. Noting that the summation in (5) is a finite geometric series, we obtain

$$C_p = \begin{cases} 1 + h p_r d_{avg} & \text{if } p_r d_f = 1 \\ 1 + p_r d_{avg} \left(\frac{1-(p_r d_f)^h}{1-p_r d_f} \right) & \text{otherwise} \end{cases} \quad (6)$$

Equation (6) validates the intuition that routing overhead C_p is directly proportional to rebroadcasting probability p_r and the number of hops that an RREQ packet has to travel. As $p_r \to 1$, routing overhead becomes a univariate function of h. Clearly, the value of h will increase with the size of the network and so does the routing overhead. For $p_r < 1$, the routing overhead will be comparatively smaller because not all nodes in the network rebroadcast RREQs. The only remaining parameter of routing overhead in (6) is the *expected forward degree*, d_f, of a node. The next section derives the expression for expected forward degree of a node.

3.1.2 Derivation of Node Expected Forward Degree

Expected forward degree of a node, d_f, is dependent upon the geometrical position of the node. Nodes located near the periphery of the transmission circle of a node can cover the maximum uncovered area. For example node T in Figure 1 is the peripheral node of the source S and is likely to reach higher number of nodes in strip I than the interior nodes.

To calculate the forward degree of node T, we need to evaluate the number of nodes common to the source node S and the node T, denoted by d_{common}. Then, the forward degree of node T will be $d_{avg} - d_{common}$. Now d_{common} is a function of the overlapping region which is the area in which transmissions of both the nodes can be received correctly (see Figure 2). Area of the sector VTZ, A^s_{VTZ}, making an angle θ at the center of the circle is

$$A^s_{VTZ} = \frac{1}{2} r_0^2 \theta$$

where θ is in radians. When $\theta \rightarrow 2\pi$ radians, a case of complete overlap, A^s_{VYZ} approaches πr_0^2. We need to compute the area of the overlapping region containing black dots i.e. A_{dot}. Then the total area of the overlapping region is simply the twice of this area i.e. $A_{overlap} = 2A_{dot}$. A_{dot} can be obtained by subtracting the area of triangle VTZ, $A^t_{VTZ} = \frac{\sqrt{3}}{4} r_0^2$, from the area of sector VTZ, A^s_{VTZ}. After simplification, we obtain

$$A_{overlap} = r_0^2 (\theta - \frac{\sqrt{3}}{2}). \quad (7)$$

θ can easily be calculated by considering the right angled triangle VTW. Finally, we multiply (7) with the node density (ρ) to calculate d_{common}.

$$d_{common} = \rho r_0^2 (\frac{2\pi}{3} - \frac{\sqrt{3}}{2}). \quad (8)$$

Equation (8) shows that the number of common neighbors of any two nodes will vary directly with the node density (ρ) and the transmission radius of the nodes (r_0). Now forward degree of node T is $d_{avg} - d_{common}$ which is the maximum possible forward degree of a node within transmission range of the source node S. On the other hand, nodes lying closer to source will have approximately zero forward degree as their rebroadcasts may not be heard by any node in strip I. Hence, the *Expected forward degree* of a node within the transmission circle of the source S is approximately equal to

$$d_f \simeq \frac{d_{avg} - \rho r_0^2 (\frac{2\pi}{3} - \frac{\sqrt{3}}{2})}{2}. \quad (9)$$

Equation (9) shows that the *Expected forward degree* of a node depends upon d_{avg}, node density ρ and the transmission radius r_0. It is also important to mention that we assume a constant value of d_f at each hop of the network for the sake of simplicity. In real networks, d_f will keep decreasing for protocols with $p_r \geq 0.5$ as we move away from the source and hence routing overhead calculated through our proposed model will be higher, especially in large scale networks. However, for routing protocols with smaller values of p_r (e.g. $p_r < 0.5$), d_f will approximately remain constant.

3.1.3 Evaluation for Varying Model Parameters

To further illustrate the significance of our proposed model, we setup three networks with 1000, 3000 and 8000 nodes distributed randomly in an area of $1000m^2$ with average degrees of 11, 12 and 13 respectively. r_0, is chosen according to (3) in order to keep the graphs connected. Figure 3 plots the total number of control packets generated up to a distance of 5 hops against p_r. As expected, as p_r increases, the number of control packets generated increase exponentially because more and more RREQ receivers are now rebroadcasting the route request. An interesting observation is the

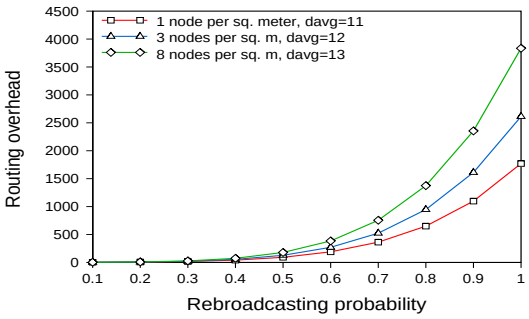

Figure 3: Routing overhead for varying rebroadcasting probabilities (p_r).

dependence of C_p on d_{avg} only. No matter what the node density is, if the average degree of the node does not change, C_p is same in each topology. This does not make sense intuitively, however. With higher node density, routing overhead must increase. But keep in mind that as node density increases, r_0 need to be decreased to keep the graph connected and hence expected increase in C_p is nullified accordingly.

3.2 Adaptation of Generic Model to Specific Protocols

3.2.1 Routing Overhead of BeeSensor

BeeSensor is a multi-path routing algorithm for wireless sensor networks based on the foraging principles of honey bees [12] with an on-demand route discovery. BeeSensor mainly utilizes three types of agents: *packers*, *scouts* and *foragers*. Packers locate appropriate foragers for the data packets at the source node. Scouts are responsible for discovering a path to an unknown destination using broadcasting. Foragers are the main workers of BeeSensor which follow a point-to-point mode of transmission and carry the data packets to a sink node.

When a source node detects an event and does not have a route to the sink node, it launches a forward scout and caches the event. A forward scout is propagated using the broadcasting principle to all the neighbors of a node. Intermediate nodes at a distance of two hops or less always broadcast the forward scout with $p_r = 1$ while rest of the nodes rebroadcast it with $p_r = \frac{1}{2}$.

When the sink node receives a forward scout, it selects a unique path ID (PID), converts the forward scout into a backward scout and returns it to the source node. Each intermediate node including the sink node associates a reward with the neighbors through which they receive the replicas of a forward scout. A backward scout is propagated back to the source node based on the reward. The sink node forwards the backward scout to the neighbor for which this reward is maximum. The process is repeated at all intermediate nodes until the backward scout is back at the source node. The source node calculates a dance number using the reported path quality and updates the routing table.

Routing overhead can be modeled by adaptation of (4) to the broadcasting principles used in BeeSensor. Recall that the nodes at a distance of two hops or less re-broadcast the forward scout with $p_r = 1$ while rest of the nodes rebroadcast the forward scout with $p_r = \frac{1}{2}$. Hence, total number

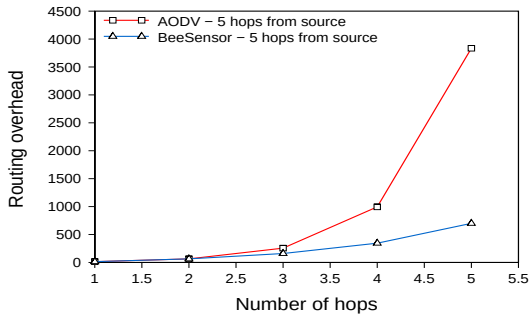

Figure 4: Comparison of AODV and BeeSensor routing overheads at different hops from the source node.

of forward scouts injected into the network or BeeSensor's routing overhead (in terms of number of forward scouts), C_p^{bee}, is given by:

$$C_p^{bee} = 1 + d_{avg} + d_{avg}d_f + \frac{1}{2}d_{avg}d_f^2 + \ldots + (\frac{1}{2})^{h-2}d_{avg}d_f^{h-1}.$$

Summing up all the $h+1$ terms and applying the finite geometric series summation, we have

$$C_p^{bee} = \begin{cases} 1 + d_{avg} + \left(\frac{h-1}{2}\right) d_{avg} \cdot d_f^2 & \text{for } h = 1 \ / \ d_f = 2 \\ 1 + d_{avg} + \\ \frac{1}{2^{h-2}} d_{avg} \cdot d_f \left(\frac{2^{h-1} - (d_f)^{h-1}}{2 - d_f}\right) & \text{otherwise.} \end{cases} \quad (10)$$

where d_f is given by (9). Equation (10) shows that as we move away from the source, the routing overhead decays exponentially (in powers of 2) with respect to the number of hops.

3.2.2 Routing Overhead of AODV

We skip the description of AODV for the sake of brevity because it is a de facto routing algorithm for ad hoc networks. However, an interested reader can refer to [7] for further details. Routing overhead model for AODV is even simpler to derive. Since each intermediate node in AODV rebroadcast the RREQ packet with $p_r = 1$, its routing overhead, C_p^{aodv}, is obtained by putting $p_r = 1$ in (6).

$$C_p^{aodv} = \begin{cases} 1 + hd_{avg} & \text{if } d_f = 1 \\ 1 + d_{avg}\left(\frac{1-(d_f)^h}{1-d_f}\right) & \text{otherwise.} \end{cases} \quad (11)$$

We have intentionally avoided the case in which intermediate nodes may generate RREPs instead of rebroadcasting the RREQs. Hence, C_p^{aodv} is the worst case AODV routing overhead. As number of hops increase, the term $(d_f)^h$ will increase, thereby resulting in higher routing overhead.

3.2.3 Comparison of BeeSensor and AODV Routing Overheads

To compare the routing overheads of both protocols, we plot C_p^{aodv} and C_p^{bee} as a function of the number of hops from the source node in a network of 8000 nodes. The results are shown in Figure 4. Initially, up to 2 hops, both protocols generate the same number of control packets as they both use $p_r = 1$ up to this hop limit. Beyond this point, BeeSensor starts rebroadcasting forward scouts selectively with $p_r = \frac{1}{2}$, thereby drastically reducing its routing overhead.

Based on these results, we conclude that BeeSensor generates significantly fewer control packets than AODV for number of hops greater than 2. While having higher routing overhead, AODV has the advantage that it will discover the shortest or optimal path to a destination. As BeeSensor does not broadcast forward scouts on all possible paths, its route optimality characteristics should be compared with AODV. The following section provides this comparison.

4. ROUTE OPTIMALITY

4.1 Generic Model

Route optimality is another key performance metric that is used for evaluation of ad hoc routing protocols. Therefore, in this section, we first propose a generic model for route optimality. This model is then adapted to the BeeSensor and AODV protocols.

We define an optimal route as a path with the least number of hops between a source and a destination. Let t denote the length of the optimal path between two nodes. We assume a dense network in which there are k edge disjoint paths between the source-destination pair under consideration. The reason for assuming edge-disjoint paths is to model the most common broadcasting pattern found in ad hoc routing protocols. Under this scheme, intermediate nodes rebroadcast the first RREQ received from one of their neighbors and discard the future RREQs. In this way, they are only likely to discover edge-disjoint paths. The model also assumes that the routing protocol only maintain a single shortest path to the destination.

We assume, as in the case of routing overhead model, that intermediate nodes rebroadcast RREQs with probability p_r during route discovery. As another generalization, we assume that a function $f[i-t]$ provides the total number of edge-disjoint paths of length i between the source-destination nodes under consideration. For instance, if there exist 10 edge-disjoint optimal paths of length t and 12 edge-disjoint paths of length $t+1$ between the source and the destination, then $f[0] = 10$ and $f[1] = 12$.

4.1.1 Probability of Optimal Path Discovery

Now if the probability of discovering an optimal path between the given node pair is ϵ, then the probability of failure (i.e. probability of not finding *any* optimal path) is $(1-\epsilon)$. Given that optimal paths are t hops long and p_r is the rebroadcasting probability, $\epsilon = (p_r)^t$. The present problem represents a Bernoulli trial with ϵ as the probability of success and $(1-\epsilon)$ as the probability of failure. Now the success probability of finding j optimal paths out of a total of $f[0]$ optimal paths is simply a binomial distribution given by the following expression.

$$b(j; f[0], \epsilon) = P(X[t] = j) = \begin{pmatrix} f[0] \\ j \end{pmatrix} \epsilon^j (1-\epsilon)^{f[0]-j},$$

where $X[t]$ is a random variable representing the number of t hop paths discovered successfully. Then the probability of

discovering at least a single optimal path is $P(X[t] \geq 1) = 1 - (1-\epsilon)^{f[0]}$. Plugging in the value of ϵ yields:

$$P(X[t] \geq 1) = 1 - (1 - (p_r)^t)^{f[0]}, \tag{12}$$

where the term $(1-(p_r)^t)^{f[0]}$ characterizes the probability that the routing algorithm fails to find any optimal path. The imperative question at this point is: What parameters minimize the *failure* probability for a given routing protocol? This minimization can be achieved in two ways: (1) By increasing the value of $f[0]$ (i.e. by increasing the number of optimal paths), or (2) by increasing the value of p_r. This argument can also be verified intuitively: for a fixed p_r, as the number of optimal path ($f[0]$) increases, the probability of failure in discovering any of the $f[0]$ optimal paths decreases. A similar argument holds for p_r and the probability of failure; note that the limiting case of $p_r = 1$ will ultimately result in a zero failure probability.

4.1.2 Probability of Suboptimal Path Discovery

Based on the derivations in the last section, we now derive the probability of discovering a suboptimal route i.e. the routes of length $t+n$ hops where $n = 1, 2, \ldots$. The question that we are trying to address is: What is the probability of discovering at least a single route of $t+1$ hops? Discovering a path of $t+1$ hops automatically implies that an optimal path of length t hops has *not* been discovered because if the optimal path is available then the suboptimal routes would be discarded by the source or intermediate nodes. Similarly, the probability that a $t+2$ hops suboptimal path is discovered assumes that no paths of lengths t and $t+1$ have been discovered. We apply this chain rule and use (12) to derive expressions for these probabilities.

$$P(X[t+1] \geq 1) = \left(1 - (1 - \epsilon p_r)^{f[1]}\right)(1-\epsilon)^{f[0]}$$

$$P(X[t+2] \geq 1) = \left(1 - \left(1 - \epsilon (p_r)^2\right)^{f[2]}\right) \times (1 - \epsilon p_r)^{f[1]} (1-\epsilon)^{f[0]}$$

$$P(X[t+3] \geq 1) = \left(1 - \left(1 - \epsilon (p_r)^3\right)^{f[3]}\right) \times \left(1 - \epsilon (p_r)^2\right)^{f[2]} (1 - \epsilon p_r)^{f[1]} (1-\epsilon)^{f[0]},$$

where $f[1]$, $f[2]$ and $f[3]$ provide the total number of available edge-disjoint paths of length $t+1$, $t+2$ and $t+3$, respectively. In a similar way, probability of finding at least a path of length $t+n$ hops is

$$P(X[t+n] \geq 1) = \left(1 - (1 - \epsilon (p_r)^n)^{f[n]}\right) \times \left(1 - \epsilon (p_r)^{n-1}\right)^{f[n-1]} \times \cdots (1-\epsilon)^{f[0]}$$

The above expression shows that as $n \to \infty$, failure probability of finding any path to the sink node approaches 1, which in turn leads to zero probability of finding any path of length $t+n$ hops. Hence, we conclude that suboptimal paths are less probable as compared to optimal paths even in case of a protocol that performs purely stochastic broadcasting of RREQs. This argument favors the use of selective broadcasting, $p_r < 1$, in ad hoc networks. Such selective broadcasting will reduce the routing overhead while optimal routes would still be discovered with fairly high probability.

4.1.3 Expected Probability of Path Establishment

Now that we have derived and compared probabilities of optimal and suboptimal path discoveries, we turn our attention to another related problem. Specifically, we note that route establishment is not always guaranteed for protocols that use $p_r < 1$. For instance, in a sparse graph, if all the nodes are not forwarding RREQ packets, there is a likelihood that no RREQ will reach the destination, and consequently a source-destination path will not be established. Therefore, in addition to the path optimality problem, it is important to evaluate the marginal probability of path establishment, irrespective of the path length.
Based on the probabilities of optimal and suboptimal paths, the expected probability of finding a path (irrespective of the path length) between a source-destination pair is:

$$E\{X\} \simeq w[0]\left(1 - (1-\epsilon)^{f[0]}\right) + \sum_{i=1}^{n} w[i]\left(1 - \left(1 - \epsilon (p_r)^i\right)^{f[i]}\right) \times \prod_{j=1}^{i}\left(1 - \epsilon (p_r)^{i-j}\right)^{f[i-j]} \tag{13}$$

where $w[i] = \frac{f[i]}{k}$ is the normalized weight of the paths of lengths i. A closer look at expressions for $P(X[t+1])$, $P(X[t+2])$, $P(X[t+3])$ and so on, reveals that the number of terms in the product continue to increase with an increase in the path length. Each new term represents a probability value which is always less than or equal to 1. When $p_r = 1$, this summation simply reduces to 1. As a result, optimal path is discovered with probability 1. For $p_r < 1$, we note that when the fractions in (13) are multiplied together, the product is a fraction which is less than the smallest fraction, $(1-\epsilon)^{f[0]}$ in this case. Thus the probability of failing to establish a route decays with an increase in the number of available paths.
In Figure 5, we plot optimal and suboptimal route discovery probabilities for varying values of p_r . The optimal path length in the figure is 5 hops, $f[i]$ is an exponentially decreasing function, and n is set to 3 in (13). In the same figure, we also plot the expected probability of path establishment, irrespective of the path length. It is clear that as p_r increases, the probability of finding an optimal route also increases, thereby resulting in a diminishing probability of finding suboptimal paths. It is also clear from Figure 5 that the success probabilities of finding suboptimal paths is very small even at very high values of p_r. In fact, we observe a very sharp decrease in these probabilities as p_r approaches 1. We have already discussed that the reason for this is the increase in the optimal route discovery probability.
Another very interesting result can be observed from Figure 5. The figure shows that the expected probability of route establishment exhibits a steady and linear increase with respect to p_r. However, the probability of route establishment reaches 0.4 at $p_r \approx 0.6$ which means that in many cases a probabilistic RREQ forwarding protocol will not even be able to establish a path to the destination. We contemplate that probably the authors of BeeSensor introduced the condition that the first two hops must always rebroadcast *forward scouts* with $p_r = 1$ to mitigate this problem. We show in the following section that this optimization significantly increases the route establishment probability. Nevertheless, this result is a clear indication that a purely stochastic flood-

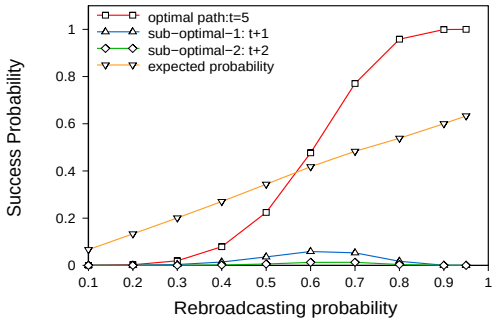

Figure 5: Probabilities of optimal and suboptimal path establishment for varying p_r.

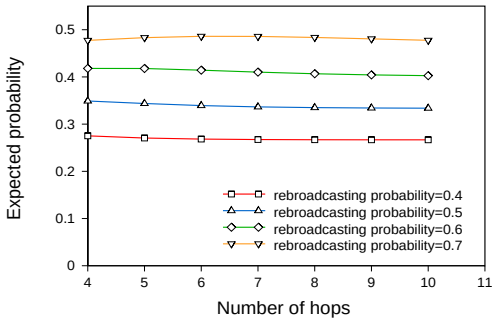

Figure 6: Expected probability of path discovery against different lengths of optimal path.

ing model is inappropriate for ad hoc routing because such an approach cannot ensure route establishment even if the rebroadcasting probability is quite high.

Figure 6 shows the expected probabilities against different optimal route lengths at different values of the rebroadcasting probability p_r. Expected probabilities seem to be a direct function of the rebroadcasting probability, p_r. Thus while the value of expected success probability increases with p_r, length of the optimal path seems to have very negligible effect on the path discovery probability.

4.2 Adaptation of Generic Route Optimality Model to Specific Protocols

We now map the route optimality probabilities to AODV and BeeSensor. Mapping to AODV is trivial because it uses pure flooding, $p_r = 1$. As we are assuming perfect channel conditions and no contention, it is guaranteed that an optimal path will be discovered i.e. $E^{aodv}\{X\} = 1$. Henceforth, we focus on evaluating the route optimality of BeeSensor.

BeeSensor uses a mix of pure and stochastic broadcasting techniques to deliver the forward scouts to the destination node. Before we talk about the route optimality of BeeSensor, recall that it is a multi-path routing protocol in which routes with better energy metric are preferred. However, for the sake of consistent comparison, we do not consider this energy metric in the definition of optimal path. Rather, we assume that all routes have equal energy and therefore the shortest path is the optimal path.

Probability of discovering an optimal route in BeeSensor is dependent upon the length of the optimal route, t. If $t \leq 3$, probability of discovering an optimal route is 1 because nodes within two hop distance of the source rebroadcast the forward scouts with $p_r = 1$. Hence, in the remainder of this section, we only model the probability of success in discovering the routes of 4 or more hops.

We go back to (13) and solve it for BeeSensor. The value of ϵ in this case will be $(p_r)^{t-3}$, where t is the length of optimal route. As individual nodes more than two hops away from the source rebroadcast the forward scout with probability $p_r = \frac{1}{2}$, approximate expected probability (by considering just three suboptimal path lengths) of discovering at least a single path using BeeSensor is

$$E^{bee}\{X\} \simeq w[0]\left(1-(1-\epsilon)^{f[0]}\right) + \sum_{i=1}^{3} w[i]\left(1-\left(1-\left(\frac{\epsilon}{2}\right)^{i}\right)^{f[i]}\right) \times \prod_{j=1}^{i}\left(1-\left(\frac{\epsilon}{2}\right)^{i-j}\right)^{f[i-j]} \quad (14)$$

Equation (14) can be analyzed by comparing it with Figure 6 which shows the expected success probability for a purely probabilistic routing protocol with $p_r = \frac{1}{2}$. We emphasize that $\epsilon = (p_r)^t$ for pure probabilistic forwarding, while in the case of BeeSensor it is $(p_r)^{t-3}$ which is a much larger value. This means that $(1-\epsilon)^{f[0]}$, the failure (of route establishment) probability of BeeSensor will be much lower than the failure probability of a purely probabilistic protocol shown in Figure 6.

5. ENERGY CONSUMPTION MODEL

Route optimality and routing overhead are the baseline metrics used for evaluation of ad hoc routing algorithms. Route optimality provides us with the success probability of finding a route of particular length, while routing overhead represents the expenditures in terms of bandwidth/energy incurred during the route discovery phase. Once we have these two metrics, we can infer a number of other performance metrics from these two baseline metrics. In this section, we use these metrics to derive simple expressions of total energy consumption to show their significance. The energy expressions only count the energy consumed during the transmission and reception of packets from the network interface, the two main sources of energy drain in an ad hoc network.

We divide the energy consumption into two broad categories. In the first category, we count the energy consumed during the route discovery process, E_{rd}. In the second category, we model the energy consumed during data transmission, E_{data}. The total energy consumed, E_{total} in joules, is $E_{rd} + E_{data}$. E_{rd} is the sum of energy consumed during the broadcasting of RREQ packets (E_{rreq}) and the energy consumed during the propagation of route reply back to the source node (E_{rrep}) i.e. $E_{rd} = E_{rreq} + E_{rrep}$.

Knowing the number of control packets generated in the network, C_p, through equation (6), E_{rreq} in joules, is given by the following expression:

$$E_{rreq} = C_p B_{rreq}(E_t + d_{avg}E_r). \quad (15)$$

Where B_{rreq} is the size of RREQ packet in bits, and E_r and E_t are the energies (in joules) required for the reception and transmission of one bit. Clearly, E_{rrep} depends on the

length of the path discovered in terms of the number of hops. If L_t is the length of the route, then E_{rrep} (in joules) is

$$E_{rrep} = L_t B_{rrep}(E_t + E_r), \tag{16}$$

where B_{rrep} represents the size (bits) of an RREP packet. Equation (16) provides the energy consumed in the propagation of a single RREP packet. Now the only remaining component is the energy consumed during the data transmission phase, E_{data}. Energy consumed in this phase is dependent upon the total number of data packets generated at the source node. If M is the number of data packets and B_{data} is the size of data packet, we have

$$E_{data} = L_t M B_{data}(E_t + E_r). \tag{17}$$

Combining equations (15), (16) and (17) gives the final generic expression for E_{total} as

$$E_{total} = L_t(MB_{data}+B_{rrep})(E_t+E_r)+C_pB_{rreq}(E_t+d_{avg}E_r). \tag{18}$$

The above expression shows that broadcast traffic is the dominant source of energy drain in an ad hoc network, especially when M is very small. C_p, for a pure flooding protocol, is theoretically equal to the size of the network while L_t is comparatively a very small number. In large scale networks, where number of nodes might be in the order of few thousands, this contribution will be prohibitive for network lifetime as each broadcast packet is received by d_{avg} nodes. Equation (18) can be easily adapted to any protocol. Size of the packets (B_{data}, B_{rrep} and B_{rreq}) might be different in different routing algorithms along with their respective routing overheads. These values can be plugged in (18) to calculate the total energy consumed by a routing protocol.

6. CONCLUSION AND FUTURE WORK

The major contribution of this paper is a probabilistic formal framework that can be used to model important performance parameters: routing overhead, route optimality and energy consumption, of ad hoc routing algorithms. We modeled a Bee-inspired routing protocol, BeeSensor, and a well known classical algorithm, AODV, with the help of our framework. The important conclusions of this undertaking are: (1) Routing overhead generated in a given network is a function of the rebroadcasting probability as well as the size of the network (number of hops), (2) In a sparse but connected network, the number of rebroadcasting nodes required to deliver the information to each node of the network is much smaller than the size of the network. This is a valuable piece of information that can help in conserving both energy and bandwidth, which are precious resources in an ad hoc network, (3) Route optimality is a function of the probability with which nodes rebroadcast RREQ packets and the total number of available paths, (4) Success probability of optimal paths discovery is always higher than the probability of suboptimal path discovery, and (5) Energy consumed during the broadcasting process is much higher in a dense network as each packet is received by a large number of receivers. As a future work, we want to refine our expression of *expected forward degree* and incorporate the contention at MAC layer so that our framework captures important features of real ad hoc networks.

7. REFERENCES

[1] C. Bettstetter. On the minimum node degree and connectivity of a wireless multihop network. In *MobiHoc*, 2002.

[2] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *MobiCom*, 1998.

[3] T. Camilo, C. Carreto, J. S. Silva, and F. Boavida. An energy-efficient ant-based routing for wireless sensor networks. In *ANTS*, 2006.

[4] H.F. Wedde et al. Beeadhoc: an energy efficlient routing algorithm for mobile ad hoc networks inspired by bee behavior. In *GECCO*, 2005.

[5] M. Farooq. *Bee-inspired Protocol Engineering: From Nature to Networks*. Natural Computing Series. Springer, (In Press).

[6] S. Kurkowski, T. Camp, and M. Colagrosso. Manet simulation studies: The incredibles. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(4):50 – 61, October 2005.

[7] C. Perkins and E. Royer. Ad-hoc on-demand distance vector routing. In *Second IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[8] C.S. Raghavendra, K.M.S. Krishna, and T. Znati. *Wireless Sensor Networks*. Springer-Verlag, 2004.

[9] M. Roth. The markovian termite: A soft routing framework. In *IEEE SIS*, 2007.

[10] M. Roth and S. Wicker. Asymptotic pheromone behavior in swarm intelligent manets: An analytical analysis of routing behavior. In *Sixth IFIP IEEE International Conference on Mobile and Wireless Communications Networks (MWCN)*, 2004.

[11] M. Roth and S. Wicker. Termite: A swarm intelligent routing algorithm for mobile wireless ad-hoc networks. In *Springer SCI Series: Swarm Intelligence and Data Mining*, 2005.

[12] M. Saleem and M. Farooq. Beesensor: A bee-inspired power aware routing protocol for wireless sensor networks. In *EvoCOMNET, LNCS 4448*, 2007.

[13] H. F. Wedde, M. Farooq, and Y. Zhang. BeeHive: An efficient fault-tolerant routing algorithm inspired by honey bee behavior. In *ANTS*, 2004.

[14] S. Zahid, M. Shehzad, S. U. Ali, and M. Farooq. A comprehensive formal framework for analyzing the behavior of nature inspired routing protocols. In *Congress on Evolutionary Computing (CEC)*, 2007.

[15] Y. Zhang, L.D. Kuhn, and M.P.J. Fromherz. Improvements on ant routing for sensor networks. In *ANTS*, 2004.