

# Balancing Traffic Load in Wireless Networks with Curveball Routing

Lucian Popa  
UC Berkeley  
popa@cs.berkeley.edu

Afshin Rostamizadeh  
New York University  
rostami@cs.nyu.edu

Richard M. Karp  
UC Berkeley  
karp@cs.berkeley.edu

Christos Papadimitriou  
UC Berkeley  
christos@cs.berkeley.edu

Ion Stoica  
UC Berkeley  
istoica@cs.berkeley.edu

## ABSTRACT

We address the problem of balancing the traffic load in multi-hop wireless networks. We consider a point-to-point communicating network with a uniform distribution of source-sink pairs. When routing along shortest paths, the nodes that are centrally located forward a disproportionate amount of traffic. This translates into increased congestion and energy consumption. However, the maximum load can be decreased if the packets follow curved paths. We show that the optimum such routing scheme can be expressed in terms of geometric optics and computed by linear programming. We then propose a practical solution, which we call Curveball Routing that achieves results not much worse than the optimum.

We evaluate our solution at three levels of fidelity: a Java high-level simulator, the ns2 simulator, and the Intel Mirage Sensor Network Testbed. Simulation results using the high-level simulator show that our solution successfully avoids the crowded center of the network, and reduces the maximum load by up to 40%. At the same time, the increase of the expected path length is small, i.e., only 8% on average. Simulation results using the ns2 simulator show that our solution can increase throughput on moderately loaded networks by up to 15%, while testbed results show a reduction in peak message load by up to 25%. Our prototype suggests that our solution is easily deployable.

## Categories and Subject Descriptors

C.2.2 [Computer-Communication Networks]: Network Protocols – routing protocols.

**General Terms:** Algorithms, Design, Experimentation, Theory.

## Keywords

Wireless Networks, Load Balancing, Geographic Routing.

## 1. INTRODUCTION

Consider a multi-hop wireless network with uniform point-to-point communication employing shortest path routing. As observed in previous work [2, 3, 25, 27], under these assumptions the center of

the network becomes “crowded”, as more paths go through the center than through the periphery of the network. This phenomenon, which we call the *crowded center effect*, implies increased congestion and energy consumption for the nodes near the center. In this paper we analyze this problem theoretically, and propose and test a practical solution to it.

Although the point-to-point communication pattern has been shown not to scale well with respect to throughput [2, 3] it is important for many applications such as wireless mesh networks [24], in-network storage and querying [23, 7, 17] or tracking [8]. Also, point-to-point communication scales better to thousands of nodes than communicating via a rooted tree. In this work, we mostly focus on the crowded center problem in a specialized and simplified setting (uniform deployment of sensors on a disc with point-to-point communication), which is a useful first step in understanding and solving the problem in its generality.

The contributions of this paper are twofold: First, we present a theoretical approach to assessing and solving the problem. We derive an exact formula for a node’s load – that is, the probability that it forwards messages on behalf of a random source destination pair – as a function of its distance from the center, when routing on shortest paths between points of a disc. This way we establish quantitatively the crowded center effect as a nearly-quadratic function peaked at the center. We then develop an approach inspired by geometric optics and based on linear programming duality that is guaranteed to find the paths (not straight-line paths, but instead shortest paths in a distorted metric, which we compute) that minimize the maximum load. We also implement approximation algorithms for finding these paths and routing along them. Unfortunately, the optimal trajectories found this way are not expressible by closed form formulas, and as a consequence this approach may not lead to a practical solution.

Our second contribution is a practical solution whose performance is very close to the optimum. Our algorithm, which we call *Curveball Routing*, works on a modified metric, obtained by projecting all nodes on a sphere. We then route greedily on the new, virtual coordinates. This algorithm is very simple, performs well in practice, and has similar computational complexity to the two-dimensional greedy geographical routing. We test our solution both by simulation and on a wireless testbed.

Throughout this paper we assume that nodes have location information and use geographical routing, a case which has some important advantages such as mathematical simplicity, scalability, and low overhead.

This paper is organized as follows. In Section 2 we present the crowded center problem and provide an exact formula for the load distribution. In Section 3 we present a theoretical approach for

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Mobihoc '07, September 9–14, 2007, Montréal, Québec, Canada.  
Copyright 2007 ACM 1-59593-684-4/07/0009...\$5.00.

solving the crowded center problem. In Section 4 we present Curveball Routing, a practical heuristic approach, while in Sections 5, 6, and 7 we present high level simulations, ns2[11] simulation results, and performance results from a real deployment. Related work is presented in Section 8, while we present our conclusions in Section 9.

## 2. THE CROWDED CENTER EFFECT

In this section we describe our assumptions on the communication model, and capture the crowded center effect by an explicit formula for the load of a node as a function of its distance from the network center.

### 2.1 Communication Model

We assume a set of uniformly distributed nodes within a specified planar geometric shape, assumed to be a disc of radius  $R$ . Nodes can communicate with their neighbors situated within a fixed communication range (the wireless unit disc model). Nodes are aware of their own location as well as their neighbors' locations. In *greedy routing* (also known as geographic routing) the forwarding layer sends a packet to a destination node with location  $L$  to its neighbor that is closest to  $L$ . The locations used for routing can be physical locations (e.g. by GPS) or virtual positions such as NoGeo[4].

We also assume a point-to-point communication pattern where the endpoints of each transmission are selected uniformly at random from the nodes of the network.

Finally, we are not concerned with greedy fallback modes such as perimeter routing [20] or CLDP [21]. These are orthogonal to the following discussion and do not qualitatively affect it.

### 2.2 Load Probability Density

In random point-to-point transmissions, different nodes in the network will forward packets for flows pertaining to different communication endpoints. The *load probability* of a node is the probability that it participates (either as endpoint or intermediary) in a communication between two randomly selected nodes. The determining factor for a node's load probability is its position inside the network: intuitively, the "center" of the network will take part in more communications than the "periphery" of the network and thus its load probability will be higher. This leads to congestion and more energy spent in the central region than at the edge nodes. We quantify this for the case of a disc-shaped network.

We approach the analysis of greedy routing by looking at the network as a *continuous* space (a disc) rather than consisting of discrete nodes. This is consistent with our view of a high number of uniformly deployed nodes. Hence we now have a load probability density instead of a finite set of values. Greedy routing follows a zig-zag path from source to destination; as the network becomes more and more dense, these zig-zag paths get closer and closer to straight lines; accordingly, we assume that the communication between two points on a continuous disc proceeds by a straight line. This continuous model should predict the statistical behavior of large discrete networks (and this is confirmed by our simulations). Note that we implicitly assumed the communication range to be much smaller than the network diameter.

Theorem 1 gives a formula for the load probability density under the continuous model of greedy routing. The intuition behind the desired formula is this: Consider a node and a greedy path going through it. Intuitively, the probability that this node will participate

in a communication along this greedy path is proportional to the product of the number of nodes on the path on either side of the node. Going now to the continuous case, the probability that a node participates in some straight-line path along a chord of the disc going through it is related to the product of the lengths of the segments of the chord on each side of the point. For a disc, this product is precisely  $R^2 - r^2$ , where  $R$  is the radius of the disc and  $r$  is the distance from the point to the center of the disc. However, this intuitive calculation is not exact, because it underestimates longer-range communication; the precise formula is slightly different.

**Theorem 1** In a uniform disc network of radius  $R$  with random point to point communication, the load probability density (probability to participate in a communication) for a node at distance  $r$  from the center is proportional to

$$(R^2 - r^2) \cdot \int_0^{2\pi} \sqrt{R^2 - r^2 \cdot \cos^2(\theta)} \cdot d\theta$$

**Proof:** Due to space constraints we omit the proof, which is based on a rather elaborate calculation and a coordinate transformation for computing integrals on stripes going through a small disc of radius  $\epsilon$  centered at the desired point. For the proof see [22]. ■

The integral occurring in Theorem 1 is an incomplete elliptic integral of the second kind, which is not expressible in closed form. Fig. 2 shows the shape of the formula, normalized to be 1 at the center. A more elaborate calculation was independently performed in [26] for the density of nodes following random polygonal trajectories within a region. Instantiating the computations of [26] to a disc of radius  $R$  we obtain a formula identical to that given by Theorem 1, although the proof is different. Note further that the authors of [25] also propose a formula for a similar load measure, resulting in a function proportional to  $(R^2 - r^2)^2$ . As we will show, our formula predicts better the load for large networks.

The crowded center effect is now clear: nodes closer to the center route a lot more messages than nodes at the perimeter. In the rest of this paper we show that one can relieve the central congestion without creating any new hotspots by using curved paths that avoid the center. However, curved paths must come at the expense of increased total network load, since shortest (straight-line) paths do minimize the total load.

What is the smallest maximum load that can be achieved by curved paths? The average of the formula in Theorem 1 over all points of the disc is 0.45. It is easy to see that this is an unachievable lower bound for the maximum load, because the maximum cannot be smaller than the average, and every algorithm except shortest path routing has an average load higher than that of shortest path routing.

## 3. COMPUTING THE OPTIMUM

In this section we formalize the problem of minimizing the maximum load and show a rigorous connection between the optimal routing paths and the curved paths followed by light in a medium where the index of refraction varies as a function of the distance from the center. We also outline algorithms for approximating the optimal paths and routing on them. We focus on a disc shaped network; however, we work in the more general framework in which the communication pairs are drawn from any rotationally symmetric distribution.

### 3.1 Problem Formulation and Analysis

A probability density function  $f(p,q)$  over pairs of points in the 2-dimensional unit disc is *rotationally symmetric* if it depends only on the distances of  $p$  and  $q$  from the origin and the central angle between them. Intuitively,  $f(p,q)$  is the probability that there is a communication between points  $p$  and  $q$ . For each unordered pair of points  $(p,q)$  we are to specify a *flow* between  $p$  and  $q$ ; i.e., a probability distribution over a finite set of continuous paths between  $p$  and  $q$ .

Given such flows, the *load* in any region  $A$  within the unit disc is the expectation of  $X(A)/\text{Area}(A)$  where  $X(A)$  is a random variable defined by the following experiment:

1. Draw a pair  $(p,q)$  from the probability distribution defined by the density function  $f$ ;
2. Draw a path from the probability distribution of paths between  $p$  and  $q$ ;
3.  $X(A)$  is the length of the intersection of the chosen path with region  $A$ .

The load at a point  $p$  is the limit of the loads of an infinite sequence of nested neighborhoods whose intersection is the point  $p$ . By the Radon-Nikodym Theorem of measure theory, this limit is the same for all such nested sequences of neighborhoods. Our problem is to specify the flows so as to minimize the maximum load in the disc.

**Theorem 2** Let  $u(r)$  be a nonnegative continuous function for  $r \in [0,1]$ . Let the *cost* of any path be the integral of the function  $u(r)$  over the path. The optimal solution (flow for every pair of points) for minimizing the maximum load is characterized by such a function  $u(r)$  and the following two properties:

1. If a path is included in the flow between  $p$  and  $q$  then it is of minimum cost among paths between  $p$  and  $q$ ;
2. If  $z$  is the maximum load over all points in the disc, then, for all  $r$  such that  $u(r) > 0$ , the load is equal to  $z$ .

**Proof:** A sketch of the proof can be found in Appendix A. ■

To understand the theorem, think of the quantity  $u(r)$  as the cost per unit length of a path at a point whose distance from the center is  $r$ . This establishes a link with geometric optics, where the *index of refraction* of a medium is defined as the speed of light in a vacuum divided by the speed of light in the medium. Fermat's Least-Action Principle states that, given the index of refraction as a function of position, a ray of light between a source and a detector follows a minimum-time path.

Thus our problem is isomorphic to the following problem in optics: given a rotationally symmetric probability distribution over the locations of source-destination pairs in a disc, choose the index of refraction as a function of radius to minimize the maximum expected intensity of light.

There is an alternative *economic* interpretation of our result: Imagine a large disc-shaped city in which authorities have imposed a mini-toll for any vehicle passing by any intersection; the toll depends on the intersection. Drivers will be expected to follow paths that are cheapest under the toll regime. Our theorem essentially states that, in order to minimize the maximum congestion, the tolls should be a function of the distance of the city center; and any intersection which is not congested should be free.

### 3.2 Computing Trajectories given $u(r)$

Given a function  $u(r)$ , define a *geodesic* between  $p$  and  $q$  as a path between the two points that minimizes the path integral of  $u(r)$ . Using polar coordinates, a path between  $p = (r(p), \theta(p))$  and  $q = (r(q), \theta(q))$  is specified by a continuous function  $r(\theta)$ , defined for  $\theta$

in the interval  $[\theta(p), \theta(q)]$ . The *path integral* is equal to:

$$\int_{\theta(p)}^{\theta(q)} u(r) \sqrt{r^2 + (dr/d\theta)^2} d\theta \quad [\text{eq. 3.1}], \text{ where } r = r(\theta). \text{ The Euler-}$$

Lagrange differential equation of the Calculus of Variations specifies in this case that a geodesic must satisfy the differential equation:  $dr/d\theta = r \sqrt{u(r)^2 r^2 / c^2 - 1}$  [eq. 3.2]. Here  $c$  is a free parameter, and the solution is defined only for  $c \leq r \cdot u(r)$ .

### 3.3 Approximating the optimal cost function

We have implemented an algorithm to approximate the optimal cost function  $u(r)$ . We use a discrete network and associate costs to nodes. The cost of a link joining two nodes is the product of its length, and the average cost of its endpoints. We start with equal costs and at each step we compute minimal cost paths between nodes using Dijkstra's algorithm. We then compute the load of each node as the number of node-to-node shortest paths that contain it; we then modify node costs by decreasing those that have smaller than average loads, and increasing those with higher than average load (the reader familiar with linear programming will recognize this as a dual modification step of a primal-dual algorithm; the linear program is given in Appendix A). We repeat this process until the load converges to a common value at all nodes within an acceptable tolerance. Finally, we use polynomial interpolation to determine a function  $u(r)$  that best matches the discrete set of node costs. Our algorithm converges to a function that is approximated well by the following polynomial:

$$u(r) = 1.8 * r^3 - 3.1 * r^2 + 2.3$$

### 3.4 Routing on Optimal Paths

Unfortunately, we cannot compute a closed form formula for the optimal trajectories because the path integral can be symbolically computed only for a few particular cost functions.

However, we can still route along approximately optimal paths by computing numerical approximations at nodes in the network. This is done by finding a solution to eq. 3.2. More exactly, it can be shown that optimal trajectories can at most have a minimum radius point between the two endpoints, and thus we need to find  $c$  and  $r_{min}$  that satisfy  $\int_{r_{min}}^{r_{max}} dr / r \sqrt{u(r)^2 r^2 / c^2 - 1} = \theta_f$  [eq. 3.3], where  $\theta_f$

is the central angle between the start and end nodes and  $r_{max}$  the maximum between the start and end radii. To compute the constant  $c$  we designed an adaptive search algorithm, using as feedback the error in estimating  $\theta_f$ . If the minimum radius along the optimal trajectory  $r_{min}$  is not at one of the endpoints, the integral in eq. 3.3 is replaced with the sum of two integrals, one from  $r_{min}$  to  $r_{start}$  and one from  $r_{min}$  to  $r_{end}$ . It can be shown that  $c = r_{min} u(r_{min})$ , and we compute  $r_{min}$  for the current  $c$  by solving this equation. After matching eq. 3.3 by approximating  $c$  and  $r_{min}$  we then find  $dr/d\theta$  and compute the absolute direction that the packet should take.

Having the instantaneous direction, we can forward the packet towards the neighbor that deviates least from it. This approach would result in a small deviation from the geometric trajectory but in a significant overhead in the number of hops and thus in load. To approximate optimal paths with a smaller number of hops, we routed the packet to the neighbor that made most geometric progress towards a point projected on the instantaneous direction at the radio range distance. We implemented this algorithm and we present simulation results in Section 5.

Nevertheless, this approach to routing is not really practical since it involves a search algorithm with numerical integrals at least at the first hop but most likely at some intermediary nodes too as the trajectory might deviate during routing.

We also envisioned an algorithm that allows routing on optimal trajectories without requiring an integral calculation. The algorithm splits the disc into discrete rings and restricts attention to piecewise-linear trajectories that change direction only at ring boundaries. This approach does not need to compute integrals or roots for  $u(r) \cdot r$ , but uses large pre-computed tables and gives a rougher approximation. Due to lack of space we do not elaborate.

## 4. CURVEBALL ROUTING

The algorithms in the previous section are of theoretical importance, showing that we can approximate the minimum max load, but, as we have mentioned, they require either large pre-computed tables or excessive computational overhead. In this section we present an alternative heuristic solution that achieves comparable results in a much simpler way.

Imagine that nodes are deployed on the surface of a three dimensional sphere instead of a planar shape. If nodes communicate only on the sphere surface and communication is uniform, the crowded center effect vanishes due to symmetry (there is no such thing as a “center” on the sphere surface). Thus, a natural idea is to map the two-dimensional disc on a sphere and route greedily on those virtual coordinates.

### 4.1 The Sphere Mapping

It is obvious that we cannot construct a distance-preserving mapping from the disc onto the sphere. However, we shall now propose a heuristic mapping yielding excellent results.

Fig. 1 shows how we do the mapping. The sphere is intersected at its equator by the plane where the network is situated. We pick a point on the polar axis, and connect it to all the nodes in the network. In our experiments we used the pole as the projection point (other projection points gave similar or worse results). The point where each line intersects the sphere will be the new (virtual) coordinate of that node. Note that this mapping preserves topological neighborhoods, thus a node’s neighbors will still be found in its vicinity and paths will remain continuous.

For a better mapping, we can first apply a power function to the initial coordinates, i.e., a function that leaves the angle unchanged and replaces the radius  $r$  of each point by  $r^\alpha$  for a constant  $\alpha$ , and projects these modified coordinates on the sphere.

### 4.2 Routing

In Curveball Routing we route greedily to the neighboring node that yields the most geometric progress towards the destination in the spherical coordinates. If routing on these 3D coordinates fails, we fall back to 2D greedy routing. If this fails, any of the existing fallback methods (e.g. perimeter [20], CLDP [21]) can be applied.

### 4.3 Mapping Limitations

The described mapping has some undesired properties. First, it is not uniform: nodes situated at the bottom of the sphere in Fig. 1 will be sparser, and the mapping becomes denser as we advance towards the sphere top. This can partly be improved by applying an initial modification to the coordinates such as the power transformation explained above.

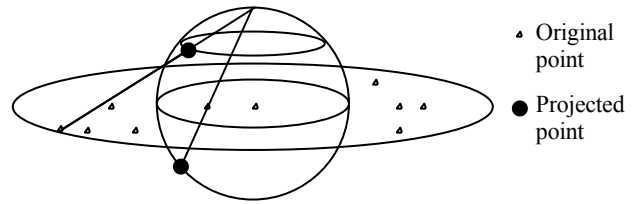


Fig. 1 Projecting the disc on a sphere

Second, the sphere will not be completely covered; there will be an upper cap with no points mapped to it. We can decrease the unoccupied region by reducing the radius of the sphere and/or lowering the sphere so that the network plane is no longer equatorial. However, one less desirable effect of trying to cover more of the sphere is that more trajectories will attempt to traverse the boundary of the upper cap, making it a highly loaded area. We call this the “edge effect”. On the other hand, this boundary is physically longer and, moreover, would get almost no traffic on a hemisphere mapping. We explore this tradeoff more in Section 5.

Finally, the best mapping parameters (i.e. sphere radius and power  $\alpha$ ) are somewhat dependent on the network (shape, number of nodes, radio range) as we will see in Section 5, where we also explore the performance of Curveball Routing on regions other than the disc.

### 4.4 Discussion of the method

An important characteristic of Curveball Routing is that it is easy to implement. Each node only needs to compute neighbors’ spherical coordinates and route greedily in that 3D space. These coordinates need to be learned only once and remembered. However, nodes need extra information: the center of the network and the network diameter (or their estimates). The computational overhead compared to 2D greedy routing is negligible. The communication overhead is minimal, only one bit is necessary to specify one of the two states: two dimensional routing or spherical coordinates routing.

The projection approach can be extended to work for different planar network shapes such as rectangles, where we can use a surface other than the sphere to receive the projections. But note that unlike the disc case, rectangles are harder to optimize due to the narrower cross section. Identifying the optimum projection surface appropriate for each shape is an important and challenging problem which we leave for future work. For now, we restrict ourselves to measuring how the sphere projection works on other shapes; the results are consistently good.

### 4.5 Some other heuristics

Before settling on the spherical projection we have considered other heuristics, two of which we briefly mention here.

*Modified Density:* Apply a coordinate transformation  $f(radius)$  to modify the virtual density throughout the network to compensate for the load imbalance. It is not clear whether there exists a function such that the new greedy paths (curved in reality) are the optimal paths. In any case, the computation of geodesics for a general  $f$  is complex, and a few attempts with special  $f$ ’s were unsatisfactory.

*Ring Routes:* Imagine a disc-shaped metropolitan area, in which congestion is avoided by co-centric *ring roads*. To drive from any point to another we start at the ring road closest to the start point,

Fig.2 Avg. Load-Theorem 1/Opt. Approx.(15k nodes)

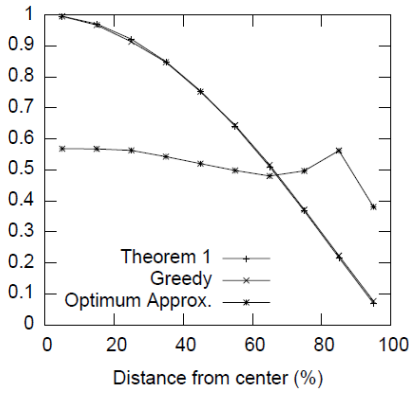


Fig.3 Max Load - Optimum Approx. (15k nodes)

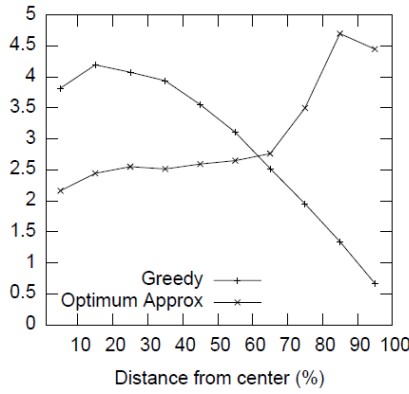


Fig.4(a) Average Load Curveball (15k nodes)

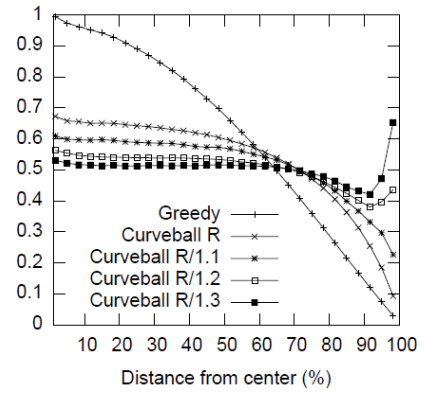


Fig.4(b) Max Load Curveball (15k nodes)

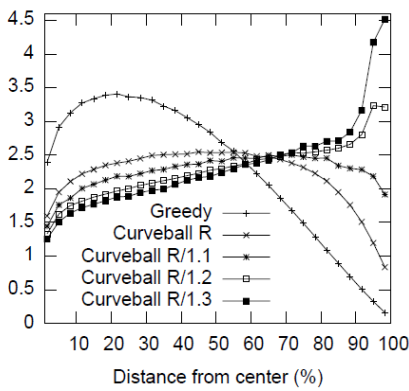


Fig.5(a) Average Load Curveball reactive (1k nodes)

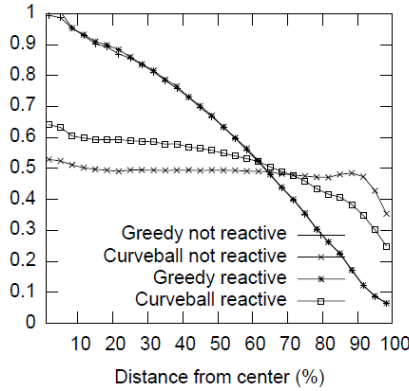
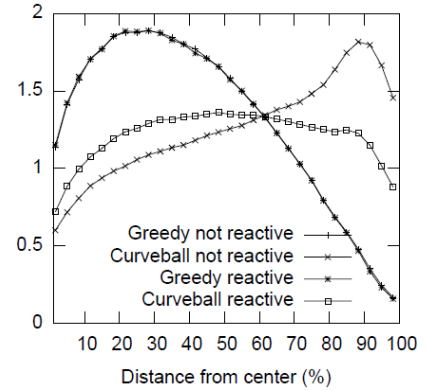


Fig.5(b) Max Load Curveball reactive (1k nodes)



end at the ring road closest to the destination, and jump between the two at a random point. Unfortunately, this intuitively appealing solution, and many variants, also proved unsatisfactory. Coincidentally, a similar idea was independently explored with some success in [27] (see Section 8).

## 5. HIGH LEVEL SIMULATION

To evaluate our algorithms we used three different environments: a high-level simulator, a low-level simulator (i.e. ns2), and a physical test bed implementation. The high-level simulator allows us to gain intuition into the algorithm's behavior by exploring a large number of scenarios and large networks. The low-level simulator allows us to evaluate our algorithm in a more realistic setting, but reduces our ability to explore the design space (e.g., it limits the network size). Finally, the test bed validates the simulation results in an experimental setting, and tests whether our solution is easily deployable.

The high-level simulator is packet level, modeling neither packet losses, nor interferences, and is implemented in Java.

### 5.1 Metrics and Methodology

We consider networks with nodes located either uniformly at random, or deployed on a uniform grid. All nodes have a circular communication range with the same radius. In each simulation, we chose one half of the nodes randomly and then paired each of them with a random node in the other half. Each pair of nodes exchanges one packet. For evaluation purposes, we divide the network into circular constant-width annuli, and within each annulus we measure

the average and the maximum number of packets handled (i.e., sent, received, and forwarded) by a node.

We use two main metrics to evaluate the performance:

- *average load*: the average number of packets handled by a node in a given annulus.
- *maximum (max) load*: the maximum number of packets handled by a node in a given annulus.

Correspondingly, from the above metrics we derive the improvement metrics: *decrease in average load* and *decrease in max load*. The two metrics are distinct due to the randomness of the deployment and are both important. The decrease in average load reflects how an entire area will be less loaded (e.g. consume less energy) while the decrease in max load reflects an algorithm's effect on the individual particularities of a deployment. The decrease in average load is the more meaningful measure because the maximum load can be reduced significantly by reactive routing, a form of local load balancing described later this section.

To better understand how the algorithms behave, we initially use a very large number of nodes (15,000) and then evaluate networks that are closer to real world dimensions (400-1000).

Unless otherwise specified, we use a communication radius resulting in approximately 20 neighbors for networks with nodes randomly located, and 8 neighbors for grid networks.

We do not use any greedy fall-back mechanism. We made this decision for two reasons. First, the density in all networks is high enough so that the failure probability of greedy routing is very small (i.e. ~0.2%), not quantitatively affecting results. Second, since there

are several fall-back algorithms proposed in the literature (e.g. [20, 21]), we wanted to factor out the impact of such decision.

The results plotted in this section are averaged over enough simulations that the confidence intervals are small, i.e., the worse case 95% confidence interval is well below 1% for the decrease in average load and 2% for the decrease in max load.

The y-axis of all plots is normalized so that the load in the central annulus for the greedy routing is 1. Unless otherwise specified, the horizontal axis shows the distance from the center measured in percents towards the edge.

## 5.2 Theorem 1 Validation

Fig. 2 compares the average load predicted by Theorem 1 to our simulation results for a 15000 node disc network. As shown by the plots, the simulation results are practically *identical* to the theoretical prediction, i.e., the integral over each annulus.

## 5.3 Optimum Paths Approximation

Fig.2 also shows the average load when routing on geodesic trajectories as described in Section 3.4 using the optimal cost function approximation from Section 3.3 for a network of 15000 nodes. The peak load decrease is around 44%.

The approximated optimal load is almost flat with a small decrease near the edge. There are two reasons for this. First and foremost, the cost function  $u(r)$  used is only approximately optimal. Secondly, the routing deviates from geodesic curves because we are working with a discrete set of nodes rather than a continuum. This effect gets worse close to the edge of the network; because we are using discrete points, the edge of the network is not “round” but “bumpy,” and as a result paths that should go very close to the edge end up further in the interior. We believe that fine-tuning the simulation by using a better approximation of the cost function would result in a flatter load distribution and improve our approximation of the optimum.

Fig. 3 plots the maximum load of a node within each annulus. As can be seen, in the case of random placement there is a considerable discrepancy between the maximum and the average loads. The reason for this discrepancy is the random node placement, which causes some “attractor” nodes forming particular geometric shapes to forward a disproportionate number of packets. The further an annulus is from the center, the more nodes, and consequently more attractor nodes, it contains, leading to a larger discrepancy between the max and the average load. This is similar to drawing values from a given distribution: the more values we draw the larger the difference between the maximum and the average. In the next section we explore reactive routing as a solution for this problem.

## 5.4 Curveball Routing Simulations

In this section, we evaluate our heuristic solution described in Section 4 over several simulation scenarios. We use the mapping method described in 4.1, and, unless otherwise specified, we use  $\alpha=1$  (i.e. project the original coordinates).

In summary, the results in this section are as follows:

- For a disc network we obtain a decrease in average load of 35-44% and a decrease in max load of around 25-40%.
- When nodes are deployed more uniformly or the network size decreases, the results improve. For a 400 node square grid the decrease in max load is as high as 42%.

- The average increase of the path length when using curveball routing is very small, only 3-8% longer than in the case of shortest-path greedy routing.
- The results improve as the number of neighbors increases, although improvements are not significant, i.e., less than 10%.
- While the sphere mapping also works for other shapes such as squares and rectangles, the improvements are smaller, e.g. for a 1000 random node square the decrease in average load is 36%, and the decrease in max load is 24%.

Note that the decrease in average load of Curveball Routing is very close to the optimum approximation (under 1%). The reason for this is the error in approximating the optimal paths mainly reflected by the approximate cost function but also by the routing based on instantaneous directions. The latter incurs in practice more hops, despite that optimal paths are on average shorter in Euclidean length than Curveball paths. On the other hand, Curveball selects hops using a greedy criterion aimed at minimizing hop count.

### 5.4.1 Impact of sphere radius

In this section, we show the average load for Curveball Routing as a function of the mapped sphere radius. Fig. 4 (a) plots the average load for a disc network with 15,000 random nodes for several cases of Curveball Routing. We consider spheres with radii ranging from  $R/1.3$  to  $R$ . The maximum average load decrease is 44%, and it is achieved for  $R/1.2$ . Note that as the radius becomes smaller, the “edge effect” described in Section 4.3 becomes prominent.

We stress that the load curves in Fig. 4 should not be quantitatively generalized to all possible networks. Results differ slightly depending on network specifics (number of nodes, neighborhood size). Our goal here is to build an intuition for the effect of the sphere size on load distribution. Below we look at more network parameters.

Unlike the average, the decrease in max load is only 27% and is achieved by using a sphere of radius  $R/1.1$ , which has a decrease in average load of 40%.

### 5.4.2 Reactive-routing

We now present a simple scheme to augment Curveball Routing or other similar techniques with respect to the increase of the max load. The idea is to find alternate hops avoiding the hotspot nodes that already forward a disproportionate amount of packets. In particular, each node monitors its neighbors’ load and if the next hop’s load is higher than a threshold relative to its own load it will try to avoid that node by trying to route through an alternative neighbor. If a suitable alternative is not found in spherical coordinates, the packet is forwarded using 2D greedy routing.

This reactive scheme is simple to implement as it only requires neighborhood information. In fact, the load information can be piggybacked in the hello messages.

Fig. 5(a) and (b) plot the average and max load in a 1000 node network. To accentuate the discrepancy between max and average we used a small mapping of radius  $R/1.3$ . In this case, the average is decreased 47% but the max only by 6%. However, using the reactive technique leads to a 36% decrease in the average load, and a 29% decrease in the max load for Curveball Routing. Note that due to longer radio links and the use of the same number of annuli, the maximum load is not found at the last (edge) annulus anymore.

The reactive scheme does not work as well for greedy routing, as shown in Fig. 5, the decrease in max load is insignificant. There are two reasons for this: first, Curveball Routing falls back to greedy

Fig. 6 Average and Max load for rectangle 2x1 (1k nodes)

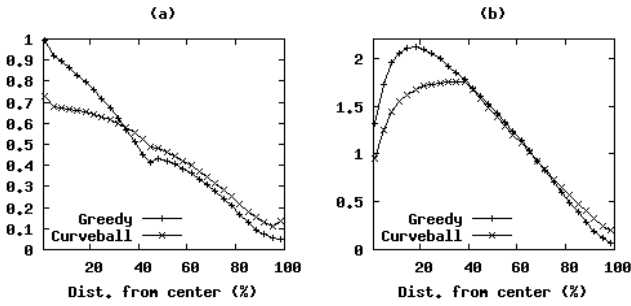
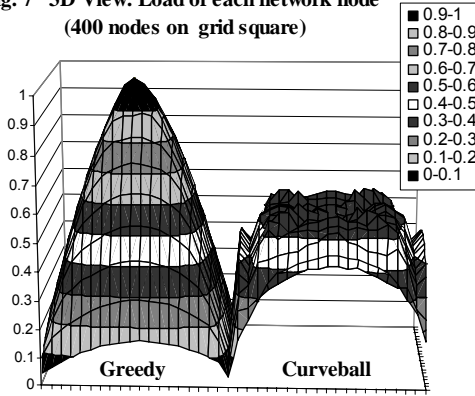


Fig. 7 3D View: Load of each network node (400 nodes on grid square)



when it cannot find a less loaded neighbor, which is not available for greedy; second, for curveball routing the hotspots are distributed across the network and thus each can be avoided individually, while, in the case of the greedy routing nodes with high loads are structurally concentrated around the network center and cannot be avoided using only local routing decisions.

### 5.4.3 Path length

In the presented simulations, Curveball Routing increases the average path length by less than 7.5% compared to the greedy paths. Similarly, the longest path increases by 59%.

### 5.4.4 Grid vs Random

As expected, if nodes are located on a grid instead of uniformly at random, the results improve. The reason is that the discrepancy between the average and the max is reduced because the grid prevents the development of isolated nodes. For example, on a 15,000 node grid, the average load decreases by 40%, while the decrease in max load is 36%. The decrease in max load is still not identical to the decrease in average load because the grid is not symmetric about the center; points on the diagonals at the same hop count from the center as points on the sides are at much larger physical distances.

### 5.4.5 The impact of neighborhood size

We studied the impact of the neighborhood size on load. Due to space constraints we only summarize the results for the same network as in Section 5.4.2 (1,000 nodes, mapping with radius  $R/1.3$ ). As expected, results get slightly worse when reducing the number of neighbors as there are fewer available paths, and avoiding the network center becomes harder. As presented, for 20 neighbors, the load decrease is 36% for the average and 29% for the max. For 15 neighbors the decrease in average is 35% and in max

Table 1. Decrease of Avg / Max (%) for various rectangles and mappings (1k nodes)

Sides ratio	$\alpha=0.9$ $r=R/1.4$ $t=3$	$\alpha=1$ $r=R/1.4$ $t=3$	$\alpha=1$ $r=R/1.3$ $t=3$	$\alpha=1$ $r=R/1.2$ $t=3$	$\alpha=0.9$ $r=R/1.4$ $t=2.5$
1 x 1	36 / 24	29 / 24	26 / 21	22 / 18	32 / 22
2 x 1	36 / 9	30 / 13	27 / 15	23 / 18	32 / 14
3 x 1	34 / 8	28 / 12	25 / 12	23 / 13	30 / 11
4 x 1	31 / 8	25 / 10	23 / 11	21 / 11	26 / 10

load 26%. Finally, for 10 neighbors the decrease in average load is 30% and in max is 22%.

### 5.4.6 Other shapes

So far we have assumed only disc networks. In this subsection we evaluate Curveball Routing on square and rectangular shapes. Note that we are still computing averages on annuli although these shapes are not symmetric to the center.

Fig. 6 shows the average (a) and max (b) behavior for a rectangle with side ratio 2:1 having 1,000 nodes uniformly located. The projection is done by first applying a power function with  $\alpha=0.9$  and then mapping on the sphere of radius equal to the half-diagonal divided by 1.4. We use the reactive scheme for routing. As expected, the results are worse than for discs: the decrease in average load is 27% and the decrease in max load is 17%.

Fig 7 shows a 3D representation of the individual node load for a 400 nodes grid network. We use the same mapping as in Fig.6. In this case, Curveball Routing decreases the load by 42%. This is equivalent to the max decrease in the rest of the charts since data is not averaged. Also due to the grid deployment there was no need for reactive routing. In addition, the increase of the path length is on average only 3%, while the maximum path length increase is 15%.

Finally, we measured results for rectangles of 1000 nodes with varying ratios between the sides and using different Curveball mappings. We present the results in Table 1. The  $t$  parameter is the reactive relative load threshold, above which nodes search alternative routes, and  $r$  is the sphere radius. First, notice that as we increase the sphere radius more traffic gets through the center and the decrease in average load gets lower while the decrease in max load increases as less traffic goes to the exterior (obviously this effect will stop when the radius becomes large enough). Also when mapping on a small sphere, setting up a smaller reactive threshold improves the max load.

### 5.4.7 Mapping Parameters

One drawback of our solution is that it does not give a precise formula to compute the parameters used to map the node coordinates (radius, power). However, in our experience, we found that “good” parameters can be determined relatively easily by performing a small number of simulations or experiments (e.g. a starting point can be: sphere radius= $R/1.3$ ,  $\alpha=1$  and reactive threshold=3). Also, as can be seen from Table 1, results do not change dramatically or unexpectedly as we explore the design space. Note that a “good” radius strikes a good balance between the average and max loads, which may depend on the targeted network degree of irregularity.

## 6. LOW LEVEL SIMULATION

In this section we evaluate Curveball Routing in the ns2[11] simulator, showing that it can increase throughput over greedy routing with up to 15%.

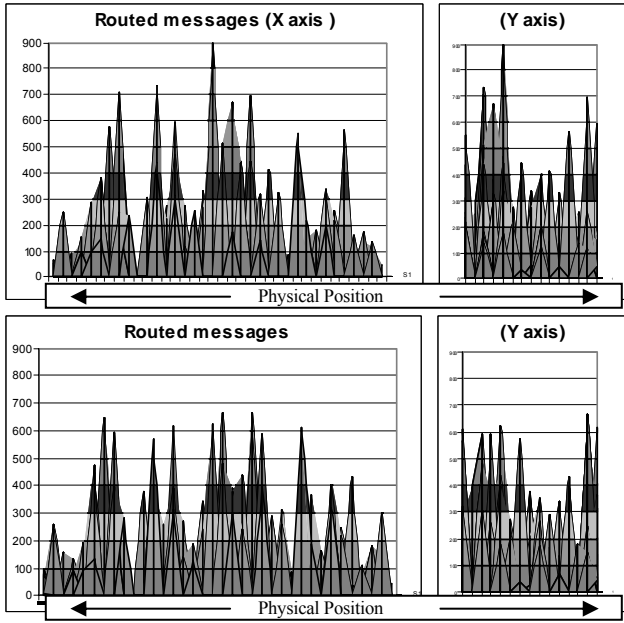
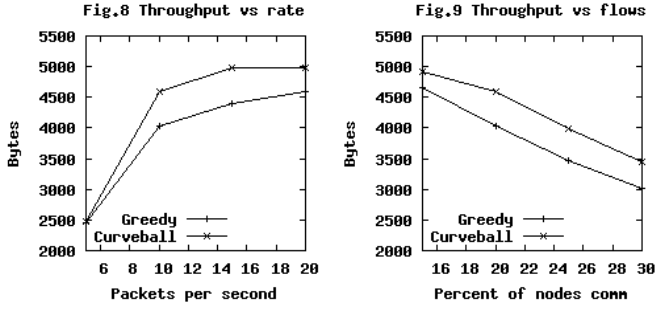


Fig. 10. Projections of the 3D load chart for the Mirage Intel Testbed

greedy = upper charts, curveball = lower charts

We use a square grid deployment of 400 nodes with 8 neighbors each. We use link level acknowledgements and 4 retransmissions.

In addition to load balancing, we are now interested in throughput which we could not measure in the high level simulator. Note that in order to get throughput increases we have to congest the network such that packets are lost due to reaching the maximum wireless capacity; otherwise we can only measure long term traffic load as before.

First, in none of our tests did Curveball Routing produce worse throughput results than the planar greedy approach. However, the relative increases are small, around 10-15%. The main reason for this is that the overall throughput increase is bounded within a constant factor of the greedy traffic (Gupta et al. show in [2] that overall network throughput is bounded by  $O(\sqrt{n})$  regardless of the routing algorithm, where  $n$  is the number of nodes). In this context, our algorithm brings throughput closer to this limit.

Fig.8 presents the overall throughput when varying the transmission rate. In Fig. 8, 20% of the nodes are communicating (40 flows). We used 5 sec. long, randomly picked, synchronized flows. The two series represent greedy routing and Curveball Routing using the same projection as in Fig.6 and Fig.7. The maximum throughput increase for Curveball Routing is a little over 13%.

Fig.9 presents the evolution of the throughput as we vary the number of flows. The transmission rate is 10 packets per second the rest of the parameters are the same as for Fig.8. The relative increase for curveball is slightly over 15%.

As an interesting side note, we noticed that the shape of the load (handled packets) in case of congestion is not flat anymore for the load balanced mapping (Fig. 7). The reason is that the central part of the network gets more congested being surrounded by interfering nodes and less packets get through compared to the edges. For this reason a slightly more imbalanced mapping (R/1.5) achieves a little higher throughput by ~2-3%. Of course, in the absence of congestion, load balancing results are identical to the ones we have presented in Fig. 7.

## 7. TESTBED EVALUATION

We implemented the sphere projection in TinyOS[12]. We present in this section experiments we have conducted on the Intel Research Berkeley deployment, Mirage[19]. This was the largest test bed we had available and consists of around 90 MicaZ nodes.

### 7.1 Methodology

We only performed tests on load balancing. We did not measure congestion because the rectangular testbed (of aspect ratio 2.5) is too narrow with respect to the communication range; nodes from the opposite sides (on the narrower dimension) interfered with each other. Another effect of the long and irregular communication range is that the load peak does not form in the center for the narrow side but closer to an edge of that side. We initiate one packet from each node towards each of the other nodes and we measure the number of messages routed by each node. We use LQI (Link Quality Indicator filled by the CC2420 radio) to select neighbors. We use 5 retransmissions and pure greedy routing (no perimeter fallback). In order to avoid congestion we use random jitters and different destination sequences at each node. Finally, because the testbed was small, we only measured the decrease in the maximum number of routed packet and we did not compute ring averages.

### 7.2 Results and Experience

We performed tests in two periods of time separated by around three months. In the first testing period, there were 93-94 (one was periodically going down) sensors operational in the testbed, in the second period 88.

Within the first period we tested only the non reactive version of our protocol. Fig. 10 presents two-dimensional projections on the two rectangle axes of the 3D load distribution in the case of greedy (upper charts) and Curveball Routing (lower charts) for these tests. The height of the column represents the number of routed messages. We obtained best results by projecting on a sphere slightly larger than half of the diagonal (half\_diagonal\*1.2). We think this is related to the long communication range with respect to the narrow side that pushes the load on the edge, which renders mapping on a smaller sphere worse.

The decrease in the maximum number of routed messages for curveball routing was 26% compared to greedy. The total number of messages increased (due to longer paths) by 4.5%. We did not include retransmissions in our counts since greedy routing uses longer links (more prone to losses) and the comparison would be unfair. Moreover, because of the shorter links, more messages reached the destinations with Curveball Routing (over 5% increase).

In the second period of testing, the pure, non reactive, algorithm performed worse than in the initial tests. The maximal load decreased by only 8% of the greedy peak. The main reason is that a part of the network center became sparser and one node with a stronger radio would take the highest peak for both methods.

We also tested the algorithm with the reactive enhancement. The reactive version of the curveball algorithm performed better dropping the max load by 13% from the reactive greedy maximum.

## 8. RELATED WORK

In [27], the authors also address the crowded center problem for a disc network and use the formula derived in [26] for the load probability, which is identical to the one derived in this paper. The authors theoretically analyze routing on inner and outer radii and propose a randomized choice between greedy and routing on the inner/outer radii to level the load. This heuristic has a smaller reported decrease of the central load than Curveball Routing and incurs a high path variance, which increases the convergence time to load balancing and also may affect other protocols. For example, two nearby sources sending to two nearby destinations each picking a different routing scheme, will get very different throughput and reliability results. In addition, the congestion feedback from proposals like [5,6] will be very different for each flow, compared to a similar feedback when using a single routing scheme.

Pham and Perreau [25] address the crowded center problem as well and also derive a formula of load probability for greedy routing. While their solution targets a closer to deployment formula, we searched for a higher level, intuitive, one, that we show is exact for large networks. They propose the use of multiple paths to mitigate the crowded center phenomenon and analytically evaluate the approach; however, the reported expected decrease is worse than that achieved by our heuristic. Also, a subsequent work [14] shows that the use of multiple paths does not achieve good results unless we use a very large number of paths.

The authors of [9] perform a theoretical study of load balanced routing in wireless networks by using multiple paths, focusing on a particular communication between fixed endpoints. A load balancing routing algorithm for very narrow wireless networks was proposed in [13].

The approach of routing to a random point has been proposed for a grid topology [18]. However this class of approaches will result in significant overhead compared to our approach considering that the best intermediary points are not random but along the optimal paths.

MAP [28] is a routing scheme that has a balancing side-effect and works for arbitrary topologies. For a disc network, MAP would route on the source's radius, a heuristic we have also tried and results in about 25% decrease in average load.

Other sensor network researchers have been inspired by Physics. In one paper, it is proposed to use electrostatic potentials to determine the appropriate sink for each node, where nodes have electrostatic charges and sinks have opposite charges [1]. Routing on optical routes has also been proposed independently in [29] to minimize a given cost function, e.g. from a variable node density. In this paper, we use a similar approach to solve the particular problem of load balancing and we provide an algorithm to compute the necessary cost function, i.e. the refraction index.

Gupta and Kumar show in [2] that the overall capacity of random wireless networks is always bounded by  $O(\sqrt{n})$  ( $n$  is the number of nodes) regardless of the routing scheme or the shape of the network.

This result implies that our overall throughput improvement can only be within a constant factor of the network size. In this paper we mostly focus on load balanced routing, which is a different problem that does not imply flow synchronization. Coincidentally, in [2] the authors use in their proof a similar mapping to Curveball Routing. Subsequent results [15] show that the capacity can be higher for constrained topologies and different receivers.

Many attempts have been made to enhance geographic routing with other information such as link quality (e.g. [10]). The same approaches can be applied to the spherical coordinates we use but we leave this for future work.

Finally, the existing wireless congestion control algorithms are reactive [5,6]. Our proactive approach can successfully coexist with them. Note that the reactive approach we propose in Section 5 to avoid load peaks for random networks should not significantly impact the reactive congestion control (it slightly varies the path).

## 9. CONCLUSIONS

We have addressed the problem of load balancing traffic in multi-hop wireless networks to increase energy usage fairness and reduce congestion. We gave a formal derivation of the load probability quantifying the crowded center problem for a disc network. Our main idea is to mitigate this effect by routing on curved paths rather than the shortest ones. We presented a theoretical approach based on geometric optics for finding and routing on the optimal paths, and developed and tested Curveball Routing, a practical approach, which routes on virtual coordinates obtained by projecting the network on a sphere. Finally, we evaluated the algorithm showing it can be successfully applied in practice.

## 10. Acknowledgement

We thank Ovidiu Savin for his valuable input on the initial stages of the paper.

## 11. REFERENCES

- [1] M. Kalantari, M. Shayman, "Design Optimization of Multi-Sink Sensor Networks by Analogy to Electrostatic Theory" IEEE WCNC, 2006
- [2] P. Gupta and P. R. Kumar. The Capacity of Wireless Networks. IEEE Transactions on Information Theory, 2000.
- [3] L. Jinyang, C. Blake, D. De Couto, H. Lee, R. Morris. "Capacity of Ad hoc Wireless Networks", ACM Mobicom 2001
- [4] Rao A. Ratnasamy S., Papadimitriou C., Shenker S., Stoica I., "Geographic Routing without Location Information," presented at ACM Mobicom, 2003.
- [5] Wan C.Y. Eisenman S.B., Campbell A.T., "CODA: Congestion Detection and Avoidance in Sensor Networks" ACM SenSys 03.
- [6] Hull B. Jamieson K., Balakrishnan H., "Mitigating Congestion in Wireless Sensor Networks," ACM SenSys, 04.
- [7] Li, X., Kim, Y. J., Govidan, R., AND Hong, W. Multi-dimensional range queries in sensor networks, SenSys 2003
- [8] Demibras, M., Arora, A., AND Gouda, M. A pursuer evader game for sensor networks. In Proc. of the Sixth Symposium on Self-Stabilizing Systems, 2003
- [9] S. Baek and G. de Veciana, Spatial Energy Balancing Large-scale Wireless Multihop Networks, IEEE INFOCOM05

- [10] K. Seada, M. Zuniga, A. Helmy, B. Krishnamachari, Energy-efficient forwarding strategies for geographic routing in lossy wireless sensor networks, ACM Sensys 2004
- [11] ns2 simulator, <http://www.isi.edu/nsnam/ns/>
- [12] TinyOs, <http://www.tinyos.net>.
- [13] Jie Gao, Li Zhang, Load Balanced Short Path Routing in Wireless Networks, IEEE Infocom 2004
- [14] G. Yashar and A. Keshavarzian, Load balancing in ad hoc networks: Single-path routing vs. multi-path routing, IEEE Infocom 2004.
- [15] P. Gupta, P. R. Kumar, "Towards an Information Theory of Large Networks: An Achievable Rate Region", IEEE Transactions on Information Theory, 2003
- [16] K. Seada, M. Zuniga, A. Helmy, B. Krishnamachari, "Energy Efficient Forwarding Strategies for Geographic Routing in Lossy Wireless Sensor Networks" ACM Sensys 2004
- [17] Shenker, S., Ratnasamy, S., Kapr, B., Govindan, R., Estrin, D. Data-centric storage in sensor networks. Sigcomm 2003
- [18] Costas Busch, Malik Magdon Ismail, Jing Xi, Oblivious Routing on Geometric Networks, SPAA 2005
- [19] Intel Mirage testbed, <http://mirage.berkeley.intel-research.net>
- [20] Brad Karp, H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks", Mobicom 2000
- [21] Kim Y.J. Govindan R., Karp B. and Shenker S., Geographic Routing Made Practical, presented at Network Systems' Design and Implementation, NSDI, 2005.
- [22] Proof for Theorem 1 listed at: [www.cs.berkeley.edu/~popa/proof\\_wireless\\_disc\\_load.pdf](http://www.cs.berkeley.edu/~popa/proof_wireless_disc_load.pdf)
- [23] Greenstein B., Estrin, D., Govindan, R., Ratnasamy, S., Shenker, S. DIFS: A distributed index for features in sensor networks. In IEEE WSNA 2003.
- [24] <http://research.microsoft.com/mesh/>
- [25] P.P. Pham and Sylvie Perreau, "Performance analysis of reactive shortest path and multi-path routing mechanism with load balance", IEEE Infocom 2003
- [26] E. Hyttiä, P. Lassila, J. Virtamo, "Spatial Node Distribution of the Random Waypoint Mobility Model with Applications", IEEE Transactions on Mobile Computing, vol. 5, no. 6, 2006
- [27] E. Hyttiä, J. Virtamo, "On Load Balancing in a Dense Wireless Multihop Network", NGI 2006, Valencia, Spain
- [28] J.Bruck, J. Gao, A. Jiang, "MAP: Medial Axis Based Geometric Routing in Sensor Networks", ACM Mobicom 2005
- [29] R. Catanuto, S. Toumpis, G. Morabito, "Opti{c,m}al: Optical/Optimal Routing in Massively Dense Wireless Networks", IEEE Infocom 2007

## APPENDIX A: Proof Sketch for Theorem 2

We consider a sequence of approximations in which we dissect the interior of the disc into cells of equal area bordered by equally spaced radial lines and concentric circles. We then approximate the total flow from one cell to another as that between the centers of gravity of the cells.

We now formulate the following linear program with an infinite number of variables and a finite number of constraints: find a flow of value 1 between each pair of cell centers so as to minimize the maximum load in any ring. The dual of this linear program has a nonnegative variable for each ring which can be interpreted as the cost per unit distance of a path intersecting the ring. These dual

variables determine a cost for any path, equal to the sum over all rings of the dual variable for the ring times the length of the intersection of the path with the ring.

The optimality conditions for the primal and its dual state:

(1) Every path used in the primal solution should be of minimum cost among all paths with the same end points.

(2) Let  $z$  be the maximum load in any ring. Then the cost per unit distance associated with a ring is positive if and only if the load in the ring is equal to  $z$ .

The primal and dual linear programs are defined as follows.

Let  $r_0, r_1, \dots, r_{t-1}, r_t$  be an increasing sequence of radii such that  $r_0 = 0$ ,  $r_t = 1$  and, for  $k = 1, 2, \dots, t$ ,  $r_k^2 - r_{k-1}^2 = 1/t$ . Let  $\Delta = (1/2\pi N)$ , where  $N$  is a positive integer. Then each cell consists of all points  $(r, \theta)$  (in polar coordinates) such that  $r_{i-1} < r < r_i$  and  $(j-1)\Delta < \theta < j\Delta$  for some  $i$  and  $j$ . Let  $C_m$  be the  $m$ th cell in some fixed ordering of the cells. Let  $S$  be the (uncountable) set of paths between cell centers and let  $S(m_1, m_2)$  be the (uncountable) set of paths between the centers of  $C(m_1)$  and  $C(m_2)$ . Let  $F(m_1, m_2)$  be the probability that out of a pair of points drawn from the distribution  $f(p, q)$  one of the points lies in  $C(m_1)$  and the other in  $C(m_2)$ . For any path  $P$  in  $S(m_1, m_2)$  let  $a(P, k)$  be  $F(m_1, m_2)$  times the length of the intersection of  $P$  with ring  $R(k)$ .

There is a primal variable  $x(P)$  for each path  $P$  in  $S$ , representing the amount of flow along path  $P$ .

The primal program is: Minimize  $z$  subject to:

$$\begin{aligned} &\text{For each } P \text{ in } S, x(P) \geq 0 \\ &\text{For each pair } m_1, m_2, \sum_{P \in S(m_1, m_2)} x(P) = 1 \\ &\text{For each ring } k, \sum_{P \in S} x(P) a(P, k) \leq z \end{aligned}$$

The dual program is: Maximize  $\sum_{m_1, m_2} v(m_1, m_2)$  subject to:

$$\text{For } k = 0, 1, \dots, t-1, u(k) \geq 0, \sum_{k=0}^{t-1} u(k) = 1$$

$$\text{For each } m_1, m_2 \text{ and } P \text{ in } S(m_1, m_2): v(m_1, m_2) \leq \sum_k u(k) a(P, k).$$

Given a feasible solution to the primal any choice of nonnegative variables  $u(k)$  summing to 1 can be extended to a feasible solution to the dual. The following complementary slackness conditions are necessary and sufficient for the optimality of a feasible primal solution  $\{x(P)\}$  and a feasible dual solution  $\{u(k)\}, \{v(m_1, m_2)\}$ :

$$\text{If } P \in S(m_1, m_2) \text{ and } x(P) > 0 \text{ then } v(m_1, m_2) = \sum_k u(k) a(P, k)$$

$$\text{If } u(k) > 0 \text{ then } \sum_{P \in S} x(P) a(P, k) = z.$$

The first condition states that flow can occur only on minimum-cost paths. The second condition states that the maximum load occurs in every ring  $R(k)$  such that the price  $u(k)$  is positive.

The Main Theorem asserts that the same conditions hold in the limit as cells shrink to points. ■