Oblivious Low-Congestion Multicast Routing in Wireless Networks *

Antonio Carzaniga[1] Koorosh Khazaei[1] Fabian Kuhn[1,2]
antonio.carzaniga@usi.ch koorosh.khazaei@usi.ch kuhn@informatik.uni-freiburg.de
[1] Faculty of Informatics, University of Lugano, 6904 Lugano, Switzerland
[2] Department of Computer Science, University of Freiburg, 79110 Freiburg, Germany

ABSTRACT

We propose a routing scheme to implement multicast communication in wireless networks. The scheme is oblivious, compact, and completely decentralized. It is intended to support dynamic and diverse multicast requests typical of, for example, publish/subscribe and content-based communication. The scheme is built on top of a geographical routing layer. Each message is transmitted along the geometric minimum spanning tree that connects the source and all the destinations. Then, for each edge in this tree, the scheme routes a message through a random intermediate node, chosen independently of the set of multicast requests. The intermediate node is chosen in the vicinity of the corresponding edge such that congestion is reduced without stretching the routes by more than a constant factor. We first evaluate the scheme analytically, showing that it achieves a theoretically optimal level of congestion. We then evaluate the scheme in simulation, showing that its performance is also good in practice.

Categories and Subject Descriptors:
C.2.2 [Computer-Communication Networks]:
Network Protocols–*routing protocols*
F.2.2 [Analysis of Algorithms and Problem Complexity]:
Nonnumerical Algorithms and Problems—*routing and layout*

General Terms: Algorithms, Performance

Keywords: geographic routing, multicast, congestion, stretch

1. INTRODUCTION

Some modes of communication are inherently multicast, in the sense that they induce the transmission of a single message to multiple destinations. This is the case of publish/subscribe communication, where each message is transmitted from the sender (the publisher) to the set of receivers that are interested in that message (the subscribers). Furthermore, while some multicast services are based on a few and relatively stable multicast groups (e.g., video streaming over IP multicast) and therefore work well with stable routing state, others are more demanding and more dynamic. For example, in *content-based* publish/subscribe, subscriptions may partially overlap, forming a large number of implicit groups—potentially, a different one for each message.

In this paper we consider a generic multicast primitive in which each message may induce a unique multicast request (m, s, T). This primitive allows a source node s to send a message m to a set of target nodes T. In particular, we consider this primitive within a wireless network. Our goal is to implement such a communication primitive through a routing scheme that is oblivious, compact, low-stretch, low-congestion, and also practical.

The scheme we propose is *oblivious* in the sense that how a request is routed does not depend on the set of requests and how the other requests are routed. We in fact prove that the scheme offers the best possible performance guarantees even in the presence of adversarial requests. The scheme is *compact* in the sense that it requires only limited state at each node, typically $O(\text{polylog}\, n)$ bits in a network of n nodes. The scheme is *low-stretch*, in the sense that the length of each path from a source to a target node, which roughly corresponds to the latency of each delivery, is optimal up to a small constant factor. The scheme is also *low-congestion*, in the sense that, for any given set of multicast requests, the maximum amount of traffic crossing a node is only a factor of $O(\log n)$ worse than with an ideal routing specifically optimized for that set of requests. Notice that this $O(\log n)$ factor for congestion is optimal for any oblivious scheme [5, 20], even for routing on 2-dimensional meshes. Lastly, the scheme is *practical* in the sense that the theoretical asymptotic behavior of the scheme can be realized in practice with good pre-asymptotic performance and small constants.

The scheme we propose is built on top of a geographical routing service whereby a message can be addressed to a given geographical location and therefore can be delivered, possibly through multiple hops, to the node that is closest to that location. Such geographical schemes exist and are compact and achieve low-stretch both theoretically and in practice [17]. The choice of a geographical communication primitive implies that, in its most basic form, the routing scheme we propose is *name dependent*. This means that nodes must be identified by some kind of address dictated by the communication layer (in this case, the node's geographic coordinates). However, it is also possible to extend such a basic routing scheme to be name independent, by means of a lookup service that can also be implemented efficiently [1].

In summary, we start from a compact and low-stretch geographical routing substrate, which for a request (m, s, t) can deliver a unicast message m from a source s to a target destination t, and we use it to build a low-congestion oblivious multicast scheme that can serve requests of the type (m, s, T) and deliver m from a source s to a set of target destinations T. A simple way to implement such

*This work was supported in part by the Swiss National Science Foundation under grant n. 200021-122137.

MobiHoc'12, June 11–14, 2012, Hilton Head Island, SC, USA.

a multicast scheme would be to implement each multicast request (m, s, T) with a series of unicast requests (m, s, t_i) for each t_i in T. However, such a scheme incurs high congestion. Intuitively, this is the case when many destinations are close to each other, even without adversarial sets of requests and instead with sources and destinations distributed uniformly over multiple requests.

A standard way to achieve low congestion with an oblivious unicast scheme is to use randomization in what is known as Valiant's trick [29]. For a unicast request (m, s, t), first route m from s to a randomly chosen intermediate destination v, and then from v to t. However, in its basic form, this trick does not work well for arbitrary worst-case sets of requests and in particular it does not work well for multicast requests. Consider for example a request (m, s, T) in which the targets $t_i \in T$ are all clustered in a small region far away from the source s. Even with Valiant's trick, a series of (unicast) copies of m going from s to a target t_i in the cluster would induce high congestion in the small perimeter around the cluster, whereas an optimal routing strategy in that case would send one copy of m from s towards the cluster, and then it would duplicate m locally to all targets within the cluster.

The scheme we propose employs a local variant of Valiant's trick, and it does that within a routing strategy that avoids congestion in the case of multicast requests. At a high level, the scheme routes a multicast request (m, s, T) along the geometric minimum spanning tree that connects the source s and all the targets in T. Then, for each edge (u, v) on that tree, the scheme uses a variant of Valiant's trick by routing m from u to an intermediate point w_{uv} chosen randomly in the vicinity of the uv segment.

In this paper we formally define this routing scheme, we then analyze its theoretical properties, and evaluate it in practice using simulation. The theoretical analysis shows that, in terms of congestion, the scheme is competitive with an ideal (non-oblivious) scheme up to a factor of $O(\log n)$, which is known to be a lower bound for congestion in oblivious schemes. The simulation study shows that the scheme is also effective in practice, with limited congestion and stretch.

2. RELATED WORK

Compared to classic wired networks, wireless ad hoc and sensor networks behave more dynamically. As a consequence, classical link-state routing protocols are often not well-suited for wireless networks and other, more reactive routing strategies are required. A standard way to do this is to combine flooding for route discovery with some caching techniques to reuse acquired routing information [7, 12, 22, 26]. While there is an abundant literature on wireless point-to-point routing, the work on wireless multicast is much less copious. In fact, Vershney claims that wireless multicast is still an important challenge [30]. Multicast protocols for wireless networks have been suggested, for example, by Royer and Perkins [27] or by Xie et al. [31].

Since the presence of wireless communication links is inherently related to the physical placement of nodes, if available, geometric information can be a powerful tool for routing. For geographic routing, it is typically assumed that all nodes are aware of their geographical position and the source node of a message knows the location of the destination. The simplest possible way to route a message that way is to proceed greedily by always forwarding a message to the neighbor closest to the destination [28]. While greedy routing is efficient in dense average-case scenarios, it might not always reach the destination. The first proposed geographic routing protocol that is guaranteed to reach the destination is face routing [15]. The delivery guarantees of the face routing protocol come at the cost of worse behavior in well-behaved settings. Therefore greedy and face routing have been combined to obtain average-case efficient protocols with guaranteed message delivery [6, 13, 17]. All these geographic routing protocols assume that the communication network is a unit disk graph. In this paper, we extend this setting with non-uniform transmission ranges in a model similar to those proposed by others [4, 16].

To apply geographic routing, the source node of a message needs to know the location of the destination. A typical application is geocast, a variant of multicast, where all nodes in a certain geographical region have to be reached [21]. If location information of the destination is not available, geographic routing can be combined with a location service that allows to efficiently search for location information of other nodes [1, 10, 19].

All routing schemes described so far do not explicitly attempt to minimize the congestion that arises in the presence of a large number of routing requests. From an algorithmic point of view, congestion has mainly been considered in the context of oblivious routing, i.e., if each routing path is chosen independently. A seminal result by Valiant and Brebner [29] shows that in a hypercube, any permutation can be routed in $O(\log n)$ steps. The path selection is randomized and uses what is now known as Valiant's trick. Each message is first routed to a random intermediate node and from there to the destination. The technique has been applied in various other networks and in particular, it was shown by Kolman and Scheideler [14] that Valiant's trick can efficiently be used in a much more general setting. The existing work on oblivious routing culminated in a breakthrough paper by Räcke [24] that shows that there is an oblivious protocol that routes every set of routing requests with expected maximum node congestion within a logarithmic factor of the best corresponding multicommodity flow solution. In light of a lower bound that even holds for 2-dimensional meshes, this is asymptotically optimal [5, 20]. Räcke's result also applies to multicast and could also be used for our wireless network model. However, the protocol state is rather heavy-weight to set up and maintain, and the given wireless setting is amenable to specialized and much more light-weight algorithms. Most closely related to our work are two papers by Busch et al. that describe algorithms for unicast in 2-dimensional meshes [9] and for geometric networks modeling dense wireless networks [8]. For unicast, this latter algorithm [8] achieves the same asymptotic bounds as the algorithm presented here. However, we believe that our randomized scheme based on Valiant's trick is somewhat simpler and easier to use. A recent survey on oblivious routing is also due to Räcke [25]. Other papers study congestion in the context of wireless network routing, but are less related to this work [11, 18, 23, 32].

3. MODEL AND DEFINITIONS

We now formally state our assumptions about the communication network and its underlying geographic routing service.

Communication Network: We assume that n wireless network nodes are located in a bounded region in 2-dimensional Euclidean space. The nodes have unique identifiers and we denote the set of nodes by V. For simplicity, we assume that the region is a square of side length L, however, the techniques work for any "reasonable" convex region. Further, we assume that nodes are aware of their position in the plane. This can be achieved by equipping nodes with GPS devices or through some localization service. Communication in the network is characterized by two positive parameters $r_C \leq r_I$ defining communication and interference radii. Whenever two nodes u and v are at Euclidean distance at most r_C, u and v can directly communicate with each other. If two nodes u and v are within distance r_I, they can cause interference to each other. Further, we assume that there is no direct communication or even interference between two nodes at distance more than r_I. We denote the ratio between r_I and r_C by $\rho := r_I/r_C$ and typically assume ρ to be a constant (independent of n). We assume that the $L \times L$-square containing the network is reasonably densely covered by nodes. Specifically, we assume that there is a parameter $r_{\rm cov}$ such that for every point in the $L \times L$-square, there is a network node within distance $r_{\rm cov}$. We assume $r_{\rm cov}$ is relatively small, such that the requirement implies that the number of nodes is at least polynomial in L/r_I.

Geographic Routing: We assume that there is a geographic routing service in place, which nodes use for communicating with each other. More formally, a node u can send a message to an arbitrary (x, y) coordinate pair within the specified geometric region that contains the wireless network nodes (i.e., the side length L square). If a message is sent to (x, y), the routing service guarantees that the node closest to (x, y) (according to Euclidean distance) receives the message. We assume that nodes populate the complete given geometric region densely enough to enable routing on almost direct paths between all pairs of nodes. We use the following definitions:

DEFINITION 3.1 (λ-PADDED PATH). *A path $P = u_1, \ldots, u_k$ connecting coordinates (x, y) and (x', y') is λ-padded if all nodes u_i of P are within Euclidean distance at most $\lambda \cdot r_I$ from the line segment connecting (x, y) and (x', y') in the plane.*

DEFINITION 3.2 (σ-SPARSE PATH). *A path $P = u_1, \ldots, u_k$ is called σ-sparse if no disk of diameter r_C contains more than σ nodes u_i of P.*

We assume the geographic routing service induces $\lambda_{\rm pad}$-padded, σ-sparse paths for some positive parameters $\lambda_{\rm pad}$ and σ. Note that this in particular implies that the node distribution is dense enough so that there is a node at distance at most $\lambda_{\rm pad} r_I$ from every point (x, y) in the geometric region covered by the network, i.e., $r_{\rm cov} \leq \lambda_{\rm pad} r_I$. Further note that the assumption that any two nodes within distance r_C are connected implies that nodes inside a disk of diameter r_C are fully connected and therefore, paths containing more than 2 nodes in such a disk can be shortened to contain at most 2 such nodes. Hence, if a $\lambda_{\rm pad}$-padded path between (x, y) and (x', y') exists, then there is also a $\lambda_{\rm pad}$-padded, 2-sparse path between the two points.

Typically, for relatively dense average-case networks, services based on greedy routing perform best. By construction, greedy routing always gives 2-sparse paths. Further, as shown in Section 7, it also gives good, $O(1)$-padded paths. For worst-case networks, geographic routing techniques [16, 17] can be used to find an $O(1)$-sparse, $O(\lambda)$-padded path, whenever a λ-padded path exists.

4. PROBLEM STATEMENT

Multicast Routing: We consider two variants of the multicast problem. A lower level geographic and a high-level name-based variant. In both cases, we are given r multicast requests $R_1, \ldots, R_r$ where request $R_i = (m_i, s_i, T_i)$ consists of a message m_i, a source node s_i and a set T_i of k_i destinations $t_{i,1}, \ldots, t_{i,k_i}$. We assume that s_i knows m_i and T_i and the objective is for s_i to send m_i to all destinations in T_i. In the case of the *geographic multicast problem*, each destination $t_{i,j}$ is given as a coordinate pair $(x_{i,j}, y_{i,j})$ and for all $i \in [r]$, message m_i has to be sent to the k_i actual network nodes closest to $(x_{i,1}, y_{i,1}), \ldots, (x_{i,k_i}, y_{i,k_i})$. In the more standard *name-based multicast problem*, each destination $t_{i,j}$ is given as a node identifier. As usual in the context, we assume that messages m_i are large compared to the size of T_i, so that the overhead of storing all destination information in the message header is negligible [2]. The geographic multicast problem is closely related to what is generally known as geocast [21]. Unlike specifying individual destinations, typically, the destinations are given by a geographic region to which a message has to be transmitted. We note that the geographic multicast service that we present can easily be adapted to efficiently work in such a scenario. In fact, in our communication model, sending to a geographic region can be modeled by sending to a dense enough set of destinations within the area.

Congestion: As discussed in Section 3, we assume that nodes at distance at most r_I can cause interference to each other. To model congestion, we assume that whenever a node u transmits, it causes interference at all nodes within distance r_I from u. Let I_u be the set of nodes within Euclidean distance r_I from node u. Hence, whenever a node in I_u sends a message, it causes interference at node u and vice versa, whenever u transmits a message, it interferes with all nodes in I_u.

To satisfy a given multicast request $R_i = (m_i, s_i, T_i)$, message m_i has to be sent from s_i to all nodes in T_i along a subtree of the network. Given some algorithm $\mathcal{A}$, let $S_i^{\mathcal{A}}$ be the multiset of nodes that transmit message m_i in order to reach all destinations in T_i, i.e., $S_i^{\mathcal{A}}$ at least contains all the inner nodes of the tree along which m_i is sent to the destinations. Given a set of r multicast requests $R_1, \ldots, R_r$ and an algorithm $\mathcal{A}$, we define the congestion $\mathsf{cong}_u^{\mathcal{A}}$ of a node u and the maximum node congestion $\mathsf{cong}^{\mathcal{A}}$ of $\mathcal{A}$ as

$$\mathsf{cong}_u^{\mathcal{A}} := \sum_{i=1}^{r} |S_i^{\mathcal{A}} \cap I_u|, \quad \mathsf{cong}^{\mathcal{A}} := \max_{u \in V} \mathsf{cong}_u^{\mathcal{A}}. \qquad (1)$$

Our main objective will be to minimize $\mathsf{cong}^{\mathcal{A}}$. Whenever it is clear from the context, we omit the superscript $\mathcal{A}$. In order to evaluate an algorithm, we intend to compare its behavior with the best possible maximum node congestion. Let $\mathsf{cong}^{\star}$ be the maximum node congestion of an optimal routing solution for the given requests $R_1, \ldots, R_r$. Consider a rectangle $\mathcal{R}$ with side lengths $w(\mathcal{R})$ and $h(\mathcal{R})$. We define $\mathsf{cut}(\mathcal{R})$ to be the set of requests R_i, $i \in [r]$ such that $\{s_i\} \cup T_i$ contains at least one node inside $\mathcal{R}$ and at least one node outside $\mathcal{R}$. To bound the optimal congestion $\mathsf{cong}^{\star}$, we introduce the following notion:

$$\mathsf{load}(\mathcal{R}) := \min \left\{ |\mathsf{cut}(\mathcal{R})|, \frac{|\mathsf{cut}(\mathcal{R})| \cdot r_I}{w(\mathcal{R}) + h(\mathcal{R})} \right\}. \qquad (2)$$

The following lemma shows that asymptotically, $\mathsf{load}(\mathcal{R})$ is a lower bound on the best possible maximum congestion $\mathsf{cong}^{\star}$.

LEMMA 4.1. *For every set of multicast requests $R_1, \ldots, R_r$ and every rectangle $\mathcal{R}$, we have $\mathsf{cong}^{\star} = \Omega(\mathsf{load}(\mathcal{R}))$.*

PROOF. Consider a multicast request R_i for which $\{s_i\} \cup T_i$ contains at least one node inside $\mathcal{R}$ and at least one node outside $\mathcal{R}$. Further, let B be the geometric area defined by all points within distance r_I of the boundary of $\mathcal{R}$. In order to satisfy request R_i, a message has to be sent into or out of $\mathcal{R}$ and therefore at least one node in B has to transmit a message.

Consider a maximal independent set S of the graph defined by the nodes V_B that lie inside B and edges $\{u, v\}$ whenever u an v are at Euclidean distance at most r_I. Whenever a node in B transmits a message, it causes congestion at some node in S. Further, since nodes in S are within distance more than r_I, the number of nodes in S is at most $O\big(1 + (w(\mathcal{R}) + h(\mathcal{R}))/r_I\big)$. Hence, by the pigeonhole principle, for every solution for the given multicast problem, some node in S has congestion at least $\Omega(\mathsf{load}(\mathcal{R}))$. □

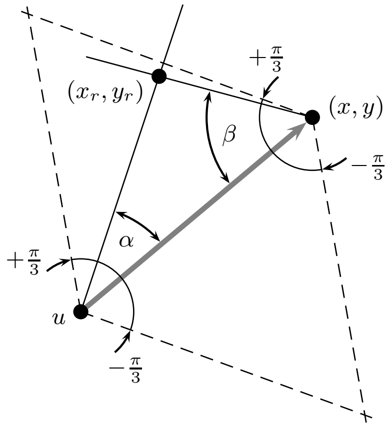

Figure 1: Choice of Intermediate Node

5. GEOMETRIC MULTICAST

Our algorithm consists of two components, which together allow to multicast a message to a set of geographical destinations based on an underlying geographic routing service as discussed in Section 3. At the core is an oblivious geographic point-to-point routing protocol with asymptotically optimal congestion properties. A multicast request is then routed on a tree by applying the point-to-point scheme.

We first describe the routing scheme to send a message from a node u to a geographical destination (x, y). The point-to-point routing algorithm is based on Valiant's classical trick of reducing overall congestion by routing messages through a randomly chosen intermediate node. To deal with worst-case collections of routing requests and to guarantee a bounded stretch factor for the routing paths, we choose the random intermediate point dependent on the source and target positions of the routing request. Specifically, a message from a node u at position (x_u, y_u) to location (x, y) is routed as follows.

1. If the Euclidean distance of (x, y) from (x_u, y_u) is at most $r_C/2$, the node closest to (x, y) is either u itself or a neighbor v of u. In that case, u directly sends the message to v.

2. Otherwise, node u chooses a random intermediate position (x_r, y_r) as follows. First, u chooses two uniform random angles $\alpha, \beta \in [0, \pi/3]$. The point (x_r, y_r) is then chosen such that the line segments from u to (x_r, y_r) and from u to the destination position (x, y) enclose an angle of α and the line segments from (x, y) to (x_r, y_r) and from (x, y) to u enclose an angle of β. There are two points (x_r, y_r) for which this is true (one to the left and one to the right of the line connecting source and destination). Node u randomly chooses one of the two points as (x_r, y_r).

   Using the underlying geographic routing protocol, the message is then routed from u to the node w closest to (x_r, y_r) and afterward from node w to the destination position (x, y).

The choice of the random point (x_r, y_r) is also illustrated in Figure 1. Note that (x_r, y_r) is chosen such that the geometric distances from (x_r, y_r) to u and (x, y) are at most as large as the distance between u and (x, y).

Based on the described scheme for point-to-point communication, we can now build the multicast routing protocol on top of it. For a given geographic mulitcast request $R_i = (m_i, s_i, T_i)$, let (x_i, y_i) be the position of the source node s_i and let $P_i = \{(x_i, y_i)\} \cup T_i$ be the set of points of the multicast request R_i. We first construct a geometric tree spanning all the points in P_i and then use the point-to-point routing algorithm to send m_i along all the edges of the constructed spanning tree. There are different ways to choose the geometric spanning tree of the points in P_i. In terms of total routing cost, the best choice would be to choose a minimum Steiner tree w.r.t. Euclidean distances. Note that the Euclidean Steiner tree problem is NP-hard. However, there is a polynomial-time approximation scheme and thus the problem can be approximated arbitrarily well [3]. Still, since we would like our algorithm to be as simple as possible, and also since asymptotically it does not make a difference, we use the Euclidean minimum spanning tree (MST) to connect the points in P_i. Such a tree can be computed locally by the sender with an efficient algorithm.

The message m_i is sent along the edges of the Euclidean MST of P_i in a straightforward manner. The tree is directed from the source s_i at (x_i, y_i) towards the destinations T_i and slightly adapted in the following way. As long as there is a directed path u, v, w such that the Euclidean distance between u and w is at most $r_C/2$, node w is attached directly to u instead of v. This allows to reach close-by nodes by local broadcast where possible.

For each (directed) edge $\big((x, y), (x', y')\big)$, a message is sent from the node closest to (x, y) to the node closest to (x', y') by using the point-to-point routing scheme described above. Assume that a node w representing a node (x, y) in the tree needs to send messages to different neighbors (x', y') in the MST. If some of the neighbors (x', y') are at distance at most $r_C/2$ from the position of w, w sends one broadcast message to all neighbors to reach the nodes closest to these tree neighbors. For all other tree neighbors, the message is sent by using the randomized point-to-point routing scheme, i.e., the message for each edge is routed via a random intermediate point as described above.

5.1 Analysis

Recall that for the analysis, we assume that for every point in the $L \times L$-square containing the network, there is an actual node within distance r_{cov}. Further when routing from a node at point (x, y) to a node at point (x', y'), the underlying geographic routing service generates λ_{pad}-padded, σ-sparse paths. For the analysis, we require a technical lemma bounding the number of local long edges of an MST in the Euclidean plane.

LEMMA 5.1. *Let T be an Euclidean MST of a set of points $X \subseteq \mathbb{R}^2$ and consider a circle $C \subseteq \mathbb{R}^2$ of radius r. The number of edges $\{p, q\}$ of T of length at least $3r$ such that $|\{p, q\} \cap C| = 1$ (i.e., edge $\{p, q\}$ connects a point inside C with a point outside C) is at most 7.*

PROOF. Consider two edges $\{p, q\}$ and $\{p', q'\}$ of length at least $3r$ such that $p, p' \in C$, $q, q' \notin C$. Let c be the center of the circle C and let θ be the angle that is enclosed by the rays cq and cq'. Let d_{cq}, $d_{cq'}$, and $d_{qq'}$ be the Euclidean distances between c and q, c and q', as well as q and q', respectively. By the law of cosines, we have

$$\cos\theta = \frac{d_{cq}^2 + d_{cq'}^2 - d_{qq'}^2}{2 \cdot d_{cq} \cdot d_{cq'}}. \tag{3}$$

Our goal is to upper bound the above expression and therefore to get a lower bound on the angle θ. Because the edges $\{p, q\}$ and $\{p', q'\}$ have length at least $3r$ and because p and p' lie in the circle C, it follows that

$$d_{cq} \geq 2r \quad \text{and} \quad d_{cq'} \geq 2r. \tag{4}$$

Since both p and p' are inside C, their distance is at most $2r$. Because $\{p,q\}$ and $\{p',q'\}$ are edges of the MST T and because we assume that their length is at least $3r$, $d_{qq'}$ has to be at least as large as the length of the longer of the two edges $\{p,q\}$ and $\{p',q'\}$. W.l.o.g., assume that $d_{cq} \geq d_{cq'}$. We then get

$$d_{qq'} \geq \max\{3r, d_{cq} - r\}\,. \tag{5}$$

We obtain an upper bound on $\cos\theta$ by maximizing the right-hand side of (3) subject to $d_{cq} \geq d_{cq'}$ and Inequalities (4) and (5). For fixed values of d_{cq} and $d_{qq'}$, the r.h.s. of (3) is a concave function of $d_{cq'}$ and is thus maximized either for $d_{cq'} = 2r$ or for $d_{cq'} = d_{cq}$.

- $\boldsymbol{d_{cq'} = 2r}$: In that case, the r.h.s. of (3) is monotonically increasing in d_{cq} and therefore maximized for $d_{cq} = d_{qq'} + r$. We then get

  $$\cos\theta \leq \frac{(2r)^2 + 2rd_{qq'} + r^2}{4\cdot(rd_{qq'} + r^2)} \leq \frac{11}{16}.$$

  The second inequality follows from $d_{qq'} \geq 3r$.

- $\boldsymbol{d_{cq'} = d_{cq}}$: In the second case, we get

  $$\cos\theta = 1 - \frac{d_{qq'}^2}{2d_{cq}^2}.$$

  The above expression gets large if $d_{qq'}$ is as small as possible and d_{cq} is as large as possible. It is maximized for $d_{qq'} = d_{cq} - r = 3r$, in which case we obtain

  $$\cos\theta = 1 - \frac{(3r)^2}{2(4r)^2} = \frac{23}{32}.$$

Combining the two cases, we therefore get $\cos\theta \leq 11/16$ which implies that $\theta > 0.812 > 2\pi/8$. □

In the following, let T_i, $1 \leq i \leq r$ be the Euclidean MST corresponding to multicast request R_i and let E_i be the directed edges of T_i, where each edge is directed away from the source s_i of R_i (i.e., in the direction in which a message has to be sent). Let $E_{\mathrm{MST}} = \bigcup_{i=1}^{r} E_i$ be the set of all directed MST edges. For a region $A \subseteq \mathbb{R}^2$ in the plane, let $E_{\mathrm{MST}}^{\nearrow}(A)$ be the set of directed edges $(p,q) \in E_{\mathrm{MST}}$ for which $p \in A$ and let $E_{\mathrm{MST}}^{\swarrow}(A)$ be the set of directed edges $(p,q) \in E_{\mathrm{MST}}$ for which $q \in A$. For each long enough edge $(p,q) \in E_{\mathrm{MST}}$, two messages are sent by using the underlying geographic routing service, one message from p to a random intermediate destination and one message from the intermediate destination to q. Let $\mathcal{M}_{\mathrm{out}}(A)$ be the set of messages sent from p to the random intermediate destination for an edge $(p,q) \in E_{\mathrm{MST}}^{\nearrow}(A)$. Further, let $\mathcal{M}_{\mathrm{in}}(A)$ be the set of messages sent from the random intermediate node to q for an edge $(p,q) \in E_{\mathrm{MST}}^{\swarrow}(A)$.

LEMMA 5.2. *Consider a square S with side length s and let v be a node at distance at least $d \geq 3s + 2r_{\mathrm{cov}} + (\lambda_{\mathrm{pad}}+1)r_I$ from S. The expected congestion at node v caused by messages in $\mathcal{M}_{\mathrm{in}}(S)$ and $\mathcal{M}_{\mathrm{out}}(S)$ is at most $O\left((\lambda_{\mathrm{pad}}+1)\cdot\sigma\cdot\rho^2\cdot\mathsf{load}(S)\right)$.*

PROOF. Let us first consider a message $m \in \mathcal{M}_{\mathrm{out}}(S)$ corresponding to some edge $(p,q) \in E_{\mathrm{MST}}^{\nearrow}(S)$. The message m is sent from the node u closest to p to a random intermediate point (x_r, y_r). Assume that the coordinates of u are (x_u, y_u). Because $p \in S$, (x_u, y_u) is at distance at most r_{cov} from S. Further, by the way the random point (x_r, y_r) is chosen, the distance from u to (x_r, y_r) is upper bounded by the distance from u to q.

Message m only causes congestion at node v if the underlying geographic routing service sends the message through a node within distance r_I from v. Because we assume that the geographic routing paths are λ_{pad}-padded, this can be the case if v is within distance $r_I(1+\lambda_{\mathrm{pad}})$ from the line segment connecting (x_u, y_u) and (x_r, y_r). Consequently, because the distance from v to S is at least $3s + 2r_{\mathrm{cov}} + (\lambda_{\mathrm{pad}}+1)r_I$, the distance between u and (x_r, y_r) and therefore also the distance between u and q needs to be at least $3s + r_{\mathrm{cov}}$. Because u is at distance at most r_{cov} from S, this implies that the edge (p,q) has length at least $3s$.

Further, recall that the line from u to (x_r, y_r) is at a random angle $\alpha \in [-\pi/3, \pi/3]$ from the line uq. Message m causes interference at node v only when α is such that the line connecting u and (x_r, y_r) passes within distance $(\lambda_{\mathrm{pad}}+1)r_I$ from v. Let β be the angle between line uv and the line connecting u with (x_r, y_r). The angle β is also a uniform random angle from some interval $[\beta_0, \beta_1]$ of length $2\pi/3$. Message m can cause interference at v if $|\beta| \leq \pi/2$ and $\ell\cdot\sin\beta \leq (\lambda_{\mathrm{pad}}+1)r_I$, where $\ell \geq 3s$ is the distance between u and v. Using $|\sin\beta| \leq |\beta|$, we get

$$|\beta| \leq \frac{(\lambda_{\mathrm{pad}}+1)r_I}{\ell} \leq \frac{(\lambda_{\mathrm{pad}}+1)r_I}{3s}.$$

Let $C_{m,v}$ be the event that message m causes congestion at v. The probability for this to happen is at most

$$\mathbb{P}(C_{m,v}) \leq \frac{2(\lambda_{\mathrm{pad}}+1)r_I)}{3s}\cdot\frac{1}{2\pi/3} = \frac{(\lambda_{\mathrm{pad}}+1)r_I}{\pi\cdot s}. \tag{6}$$

We define $X_{m,v}$ to be the random variable that counts the amount of congestion caused by m at v. Hence, $X_{m,v}$ is the number of nodes in the r_I-neighborhood of v that transmit a message while sending m from u to (x_r, y_r). Clearly $X_{m,v}$ can only be positive if the event $C_{m,v}$ occurs. In this case, the value of $X_{m,v}$ is at most $O(\sigma\rho^2)$ because we assume that the paths created by the geographic routing service are σ-sparse and a disk of radius r_I can be covered with $O(\rho^2)$ disks of diameter r_C. Let $X = \sum_{m\in\mathcal{M}_{\mathrm{out}}(S)} X_{m,v}$ be the congestion at v caused by messages in $\mathcal{M}_{\mathrm{out}}(S)$. By linearity of expectation, we have

$$\mathbb{E}[X] = O\left(\frac{(\lambda_{\mathrm{pad}}+1)r_I}{s}\cdot\sigma\rho^2\cdot|\mathcal{M}_{\mathrm{out}}(S)|\right).$$

To bound $\mathbb{E}[X]$, it therefore remains to bound the number of messages in $\mathcal{M}_{\mathrm{out}}(S)$. We have seen that each message $m \in \mathcal{M}_{\mathrm{out}}(S)$ corresponds to some MST edge (p,q) of length at least $3s$. Consider the circle C of radius $s/\sqrt{2}$ that encloses the square S. Since $p \in S$ and q is at distance at least $3s$, we have $p \in C$ and $q \notin C$. Hence, by Lemma 5.1, for each MST, there are at most 7 such edges of length at least $3s/\sqrt{2} < 3s$. Only multicast requests that contribute to $\mathsf{load}(S)$ can have MST edges with one node inside S and one node outside S. Further, for every such multicast request there are at most 7 edges in $\mathcal{M}_{\mathrm{out}}(S)$. The expected congestion at v created by nodes in $\mathcal{M}_{\mathrm{out}}(S)$ can therefore be upper bounded as

$$\mathbb{E}[X] = O\left((\lambda_{\mathrm{pad}}+1)\cdot\sigma\cdot\rho^2\cdot\mathsf{load}(S)\right). \tag{7}$$

The situation for the messages in $\mathcal{M}_{\mathrm{in}}(S)$ is almost symmetric. The messages are sent from the random intermediate destination (x_r, y_r) to a position inside S. However, the actual node sending the message might be at distance r_{cov} from (x_r, y_r), therefore we must accordingly adjust the angles for which there is congestion at node v. Instead of the value obtained in (6), the probability of $C_{m,v}$ can now be upper bounded by $\mathbb{P}(C_{m,v}) \leq \frac{r_{\mathrm{cov}}+(\lambda_{\mathrm{pad}}+1)r_I}{\pi\cdot s}$. Because $r_{\mathrm{cov}} \leq \lambda_{\mathrm{pad}} r_I$, this does not change anything asymptotically, and the congestion from messages in $\mathcal{M}_{\mathrm{in}}(S)$ can also be upper bounded by the value given in (7). The claim of the lemma therefore follows Lemma 4.1. □

We are now ready to prove the main theorem of this section, showing that the expected maximal congestion induced by our geographic multicast algorithm is within a logarithmic factor of the optimal and therefore asymptotically best possible for any oblivious algorithm [5, 20].

THEOREM 5.3. *When using the described geographic multicast algorithm to route a given set of geometric multicast requests, the expected congestion at any node v is at most*

$$O\left(\left((\lambda_{\text{pad}}+1)\cdot\log n+\lambda_{\text{pad}}^2\right)\cdot\sigma\rho^2\cdot\mathsf{cong}^\star\right).$$

PROOF. The multicast algorithm described at the beginning of Section 5 sends two kinds of messages. Most messages are messages sent through the underlying geographic routing layer. In addition, messages to local neighbors are sent by direct local broadcast. Node v can be affected by local broadcast messages only if they are sent by nodes within distance r_I from v. By adapting the MST structure and contracting paths of total length at most $r_C/2$, it is guaranteed that for each multicast request the number of local broadcast messages in each r_C-neighborhood is $O(1)$. Such messages must be sent by a node within range r_C. Hence, the total congestion at nodes within distance $r_I + r_C$ of v has to be within a constant factor of the congestion caused by local broadcast messages at v. Hence, for every multicast solution, there must be some node w close to v with congestion at least a constant times the congestion caused by local broadcast messages at v.

Let us therefore consider the congestion caused by messages that are sent through the underlying geographic routing layer. Note that all these messages correspond to an MST edge of length at least $r_C/2$ and they all either go from an MST node to a random intermediate destination or from a random intermediate destination to an MST node. We partition the $L\times L$-square containing the network into two parts, an area containing nodes close to v and an area with nodes far away from v. Specifically, we consider a square Q of side length $6r_{\text{cov}} + 3(\lambda_{\text{pad}}+1)r_I = O\big((\lambda_{\text{pad}}+1)r_I\big)$ and the area $\overline{Q}$ outside Q.

The area $\overline{Q}$ can be covered with $O\big(\log L/((\lambda_{\text{pad}}+1)r_I)\big) = O(\log L/r_I)$ squares S_i of side length s_i such that the distance of square S_i to v is at least $3s + 2r_{\text{cov}} + (\lambda_{\text{pad}}+1)r_I$ as follows. The area right around Q is covered with $O(1)$ squares of side length at most $\big(2r_{\text{cov}} + (\lambda_{\text{pad}}+1)r_I\big)/3$ such that Q together with these squares cover a larger square around v. The additional squares can be iteratively placed in the same way around the growing center square such that side length of the squares grows exponentially with the number of layers. By Lemma 5.2, for each of the squares S_i covering $\overline{Q}$, the expected congestion from messages in $\mathcal{M}_{\text{out}}(S_i)$ and $\mathcal{M}_{\text{in}}(S_i)$ is at most $O\big((\lambda_{\text{pad}}+1)\sigma\rho^2\mathsf{cong}^\star\big)$. Hence, the expected congestion from messages sent from a node in $\overline{Q}$ to a random intermediate destination and from messages sent from a random intermediate destination to a node in $\overline{Q}$ is at most

$$O\left((\lambda_{\text{pad}}+1)\cdot\sigma\rho^2\cdot\log n\right)\cdot\mathsf{cong}^\star. \tag{8}$$

Recall that we assume r_{cov} is small enough and thus the node density is large enough such that n is at least polynomial in L/r_I and thus $\log(L/r_I) = O(\log n)$.

To prove the lemma, it remains to bound the congestion from messages sent from a node in Q to a random intermediate destination or from a random intermediate destination to a node in Q. Let M be the set of such messages. Because we assume that the geographic routing service produces σ-sparse paths and because the r_I-neighborhood of v can be covered by $O(\rho^2)$ disks of diameter r_C, the congestion from each message in M is at most $O(\sigma\rho^2)$. Hence, the congestion at v from messages in M is at most $O(|M|\sigma\rho^2)$.

Every message in M corresponds to an MST edge of length more than $r_C/2$ and there are at most 2 messages in M for each such MST edge. Further, for a particular multicast request, the number of MST edges of length more than $r_C/2$ with one node in Q is linear in the number of nodes in Q and at pairwise distance more than $r_C/2$. Hence, to serve all destinations in Q, in an optimal multicast protocol, nodes in Q or within distance r_C of Q need to transmit at least $\Omega(|M|)$ times. The square Q and its r_C-neighborhood can be covered with $O\left((\lambda_{\text{pad}}+1)^2\right)$ disks of diameter r_I. Each message that is transmitted by a node inside this area causes congestion at all nodes in at least one of these diameter r_I disks. Hence, by the pigeonhole principle, some node in Q or its r_I-neighborhood has congestion at least $\Omega\big(|M|/(\lambda_{\text{pad}}+1)^2\big)$. Thus, the congestion at v caused by messages in M can be upper bounded by

$$O\left(\left(\lambda_{\text{pad}}^2+1\right)\cdot\sigma\cdot\rho^2\cdot\mathsf{cong}^\star\right). \tag{9}$$

Since the congestion caused by local broadcast messages is within a constant factor of the optimal congestion, (8) and (9) together imply the claim of the theorem. □

Remarks: If the ratio $\rho = r_I/r_C$ and the parameters λ_{pad} and σ specifying the quality of the underlying geographic routing service are constants independent of n, the statement of the theorem simplifies. The theorem shows that in this case, the maximal expected node congestion of our multicast algorithm is within a factor $O(\log n)$ of the optimal maximum node congestion. Note that it is well known that this is the best achievable bound for oblivious routing. Further, since congestion contributions from different multicast requests are independent, a standard Chernoff argument shows that the bound of Theorem 5.3 does not only hold in expectation, but also with high probability. Finally, we would like to point out that within the quality guaranteed by the underlying routing layer, our multicast protocol produces routing paths and trees that are within a constant factor of the optimal.

6. NAME-BASED MULTICAST

The multicast protocol discussed in Section 5 allows to efficiently (in terms of congestion and stretch) multicast messages if the source node of a multicast request knows the positions of all the destinations. In many cases, information about the positions of destinations is not available to the node disseminating some information. In this case, a geographic routing service can be used in conjunction with a location service that allows to query the positions of nodes [1, 10, 19]. In the following, we sketch how to apply the location service LLS [1] to our context, and we show that, if for each multicast request the destination positions can be obtained with a small number of queries to the location service, then the expected maximal congestion of looking up the destination coordinates is within a constant factor of the expected maximal congestion incurred by multicasting the messages.

Let us first briefly discuss how LLS works. We describe the most basic variant of the scheme. (The authors also present a more involved scheme that takes into account update costs when nodes are moving [1].) LLS is essentially a geometric, distributed hash table. Assume that we want to store the location information for node v with identifier id_v. We assume that there is a hash function h that assigns a coordinate $h(\text{id}_v) = \big(h_x(\text{id}_v), h_y(\text{id}_v)\big)$ in the $L\times L$-square to each node v. Using the position $h(\text{id}_v)$, we define a hierarchical tiling of the plane into squares of exponentially decreasing sizes. The corners of the squares of level $\ell = 0, 1, 2, \ldots$ of the tiling are at positions $\left(h_x(\mathrm{id}_v) + i \cdot L/2^\ell, h_y(\mathrm{id}_v) + j \cdot L/2^\ell\right)$ for integers $i, j \in \mathbb{Z}$. On every level ℓ, the position information of v is stored at the four corners of the tile that contains v. Starting from the position of v in order of decreasing levels, v's information is stored in a spiral-like fashion.

To look up the coordinate information for some node v with identifier id_v, the protocol searches in the same spiral-like fashion. Assume that node u searches for v's position information. For each level ℓ, node u queries the four corners of the tile containing u in the tiling defined by $h(\mathrm{id}_v)$. The search is done in the order of decreasing ℓ, i.e., by going from small tiles to large tiles, which forms a spiral that is shown to hit a node that stores the information about v with asymptotically optimal cost [1]. The following is a list of the most important properties of the scheme for our purposes:

1. If a node u looks up the information of some node v, the distance that has to be traversed for the search is proportional to the Euclidean distance of u and v.

2. A search for node v starting at node u follows an exponentially growing spiral. The exact paths visited during the search are determined by the position $h(\mathrm{id}_v)$. Assuming that the hash function $h(\mathrm{id}_v)$ leads to a uniformly distributed position for the origin of the coordinate system defining the tiling, it can be shown that a search from node u causes interference at a node at distance d with probability proportional to $(\lambda_{\mathrm{pad}} + 1)r_I/d$. Here, we assume that the search messages are sent through the geographic routing layer described in Section 3.

3. Assuming that the distribution of nodes is sufficiently dense, the scheme is compact. Each node only needs to store the position information of a logarithmic number of other nodes.

The next theorem shows that if at most κ look-ups are necessary for each multicast request, the expected look-up cost is asymptotically upper bounded by the expected cost for multicasting all message using our algorithm using the geometric protocol of Section 5. For the theorem, we assume that the hash function h leads to uniformly distributed positions $h(\mathrm{id}_v)$ that are independent of the given multicast requests. Due to lack of space, we only give a very rough sketch of the proof of the theorem.

THEOREM 6.1. *If each multicast request requires to look up at most κ positions, at every node v, the expected congestion caused by all look-ups is at most*

$$O\left(\kappa \cdot \left((\lambda_{\mathrm{pad}} + 1) \cdot \log n + \lambda_{\mathrm{pad}}^2\right) \cdot \sigma\rho^2 \cdot \mathsf{cong}^\star\right).$$

PROOF SKETCH. The proof follows a similar reasoning to the one in Lemma 5.2 and Theorem 5.3, where the congestion of the geometric multicast algorithm is analyzed. According to the first property of LLS listed above, a search from a node u for a node v stays within distance $O(d(u,v))$ of u, where $d(u,v)$ is the Euclidean distance between u and v. Let us therefore assume that all the κ searches of the source s_i of some multicast request R_i stay within distance $c \cdot d(s_i, t_i)$, where t_i is the destination of request R_i that is farthest away from s_i.

Let us first consider the congestion at v caused by multicast requests with a source node that is relatively far away from v. Consider a square Q of side length d that is at distance at least $2c \cdot d + (\lambda_{\mathrm{pad}} + 1)r_I$ from v. Assume that the source node s_i of multicast request R_i is inside Q. For a search of s_i to contribute to the congestion at node v, the farthest destination of R_i needs to be at least at distance $2d$ from s_i. Hence, R_i is a multicast request that has the source node in Q and at least one destination node outside Q and R_i therefore contributes to $\mathsf{load}(Q)$ of Q. By the second property of LLS described above, the probability that a search of s_i causes congestion at v is at most $O((\lambda_{\mathrm{pad}} + 1)r_I/d)$ and therefore by a similar argument as in the proof of Lemma 5.2, the expected total congestion at v from searches of source nodes in Q can be upper bounded by

$$O\left(\kappa \cdot (\lambda_{\mathrm{pad}} + 1) \cdot \sigma \cdot \rho^2 \cdot \mathsf{load}(Q)\right).$$

By Lemma 4.1, this is within a factor $O\big(\kappa(\lambda_{\mathrm{pad}} + 1)\sigma\rho^2\big)$ of the optimal maximal node congestion. As in the proof of Theorem 5.3, the congestion caused by source nodes at distance at least $3(\lambda_{\mathrm{pad}} + 1)r_I$ from v can be bounded by $O(\log n)$ times the above value because that part of the network can be covered with $O(\log n)$ squares to which the above argument can be applied. Also for the congestion from searches of sources within distance $3(\lambda_{\mathrm{pad}} + 1)r_I$ from v, a similar argument to the one in the proof of Theorem 5.3 can be applied. Together, the bounds imply the statement of the theorem. □

7. SIMULATION ANALYSIS

We now evaluate our routing scheme through simulation. This experimental analysis is intended to assess the performance of the scheme in practice, and also to characterize the effects of specific variants and parameters of the scheme itself as well as of the underlying geographical routing service. We consider three high-level research questions: (1) How does the scheme perform with various underlying routing algorithms? (2) How does the scheme perform with various selections of the random intermediate point? (3) How does the scheme perform in general under various workloads?

We first describe the implementation of the scheme and the underlying routing, and then present the simulation analysis.

7.1 Variants of the Routing Algorithms

We implemented two variants of the selection of the random intermediate point. The first variant corresponds exactly to the algorithm we describe and analyze formally in Section 5 and that is illustrated in Figure 1. This variant is parametrized by the range from which the source chooses the two random angles α and β that determine the intermediate point (x_r, y_r). In particular, we analyze the scheme when α and β are chosen uniformly in the ranges $[0, \pi/3]$, $[0, \pi/4]$, and $[0, \pi/6]$. Intuitively, wider angles would disperse traffic and therefore reduce congestion, at the expense of slightly longer paths and therefore worse total traffic.

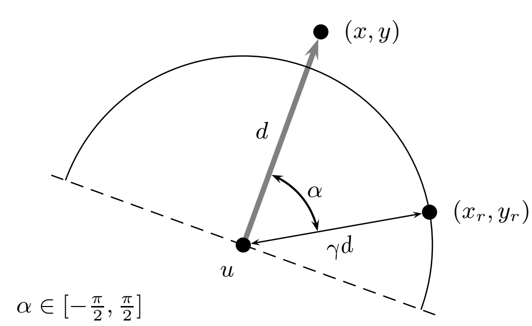

Figure 2: Alternative Selection of Intermediate Node

The second variant, illustrated in Figure 2, is a bit different: u selects an intermediate point (x_r, y_r) uniformly on a circular arc with center in u and radius $\gamma \cdot d(u, (x,y))$, where γ is a parameter of this method, and is between 0 and 2.

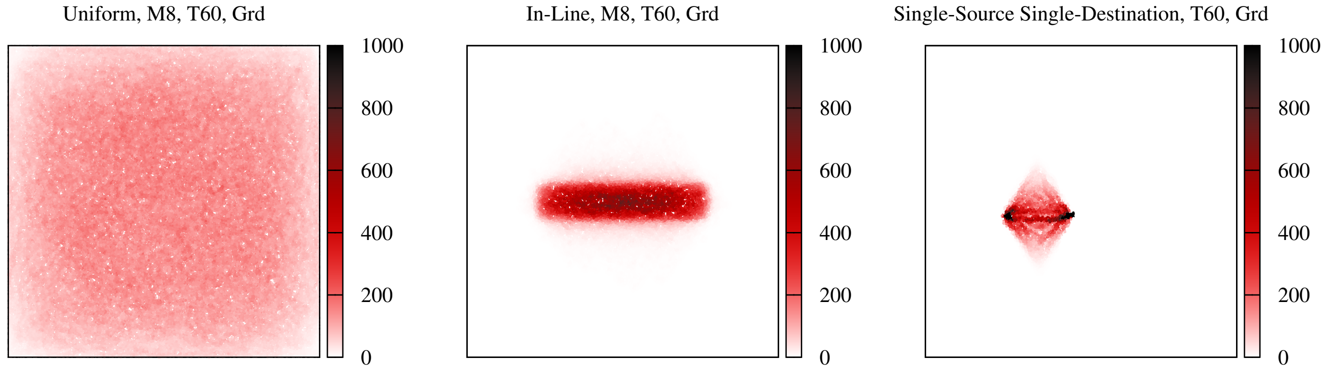

Figure 3: Examples of the Three Classes of Workloads

We also tested our scheme with various underlying routing algorithms. Recall that the geographical routing layer sends a message from a source node u to the node closest to the destination (x, y). The algorithms we considered are:

Grd Greedy routing. Each node v forwards the message to the next-hop neighbor w that is the closest to the destination (x, y).

GSP Geometric shortest path. The path between u and (x, y) is minimal in terms of geometric length.

DSP Hop-count (or "Dijkstra") shortest path. The path between u and (x, y) is minimal in terms of number of hops.

GrdRnd1 A randomized variant of greedy routing. In this case a node v forwards a message to a next-hop neighbor w chosen uniformly among the ones that advance towards the destination by at least half of the communication radius r_C.

GrdRnd2 Another randomized variant of greedy routing. A node v forwards a message to a next-hop neighbor w chosen uniformly among the ones that are within half of the communication radius r_C from neighbor $\overline{w}$, which is the closest to the destination.

7.2 Experimental Setup and Parameters

Network: We simulate a network of 80000 nodes spread uniformly over a square area of 100×100 units of length. (We also experimented with lower densities, obtaining consistent results that we do not report here for lack of space.) We set the communication radius to be equal to the interference radius ($r_C = r_I$) and we run simulations with $r_C = 1$ and $r_C = 2$ units of length. These settings correspond to a network that is dense enough to guarantee connectivity and to satisfy the more specific requirements of the underlying geographic routing, namely that it guarantees $\lambda_{\rm pad}$-padded paths for a small constant $\lambda_{\rm pad}$.

Table 1: $\lambda_{\rm pad}$ in practice

r_C	GSP	DSP	Grd	GrdRnd1	GrdRnd2
1	3.677	10.395	6.169	12.589	8.213
2	0.537	3.859	2.913	4.889	3.920

Table 1 shows the actual values for $\lambda_{\rm pad}$ for all five geographical routing algorithms. These values were computed over 10000 randomly selected paths. Notice that these are *maximum* values (as per the definition of $\lambda_{\rm pad}$-padded path) but at the chosen density the *average* distance between a routing path and a straight line between source and destination is much smaller. For lack of space, in the rest of the paper we discuss only the simulation with $r_C = 1$.

Workloads: We consider two classes of scenarios for multicast requests. One, which we denote as **uniform** in which requests involve sources and destinations chosen uniformly over the whole network, and one, which we denote as **in-line**, in which sources and destinations are chosen on a line, or more specifically on a narrow band in the middle of the network. The first class is intended to represent a generic traffic load. The second class is intended to represent a worst-case scenario for congestion. We also experimented with absolute worst-case workloads in which all requests are between the same source and the same destination. We initially show some results for all three cases for illustrative purposes, but then we focus on the **uniform** and **in-line** only because the third class is not very informative, since it incurs unavoidable congestion around the source and destination nodes.

Figure 3 shows three "heat-map" graphs representing one simulation run for each of the three classes of workloads, respectively. The graphs represent the square region covering the simulated network. Each point in the graph represents a node in the network whose color represents the total traffic (number of wireless transmissions) affecting that node, which corresponds to *load* or *congestion* of that node.

Analysis: In our analysis, we refer to a fixed set of all independent simulation parameters as a *scenario*. Thus, in a scenario we simulate all nodes running the same configuration of the geographic routing and the same configuration of our multicast routing scheme. We then simulate 1000 multicast requests, each with a fixed number of destinations chosen according to one of the scenario classes (**uniform** or **in-line**).

For each scenario we run 50 simulations to account for the variability that is due to the randomized nature of our scheme and possibly of the underlying routing. Then, for each node we compute the average load over the 50 runs, obtaining an approximation of the expected load of that node for that particular scenario. We then compute the *network congestion* as the maximum over all nodes of the per-node expected load. This is the primary metric of interest in this simulation analysis.

In summary, to answer our evaluation questions, we explore scenarios covering all combinations of the following parameters:

Intermediate point selection: type of algorithm and parameters used to select the intermediate point. We use the angle-based selection with bounds $\pi/3$, $\pi/4$, and $\pi/6$ denoted with **T60**, **T45**, and **T30**, respectively. We then use the circular-arc selection with distance multiplier $\gamma = 0, 0.5, 1, 1.5, 2$, which we denote as **C0**, **C0.5**, **C1**, **C1.5**, and **C2**. Notice that **C0** corresponds to using a deterministic straight-line routing scheme. This degenerate case is useful for comparison.

Geographical routing: type of algorithm used in the underlying routing layer. We use the algorithms described in Section 7.1, denoted as **GSP**, **DSP**, **Grd**, **GrdRnd1**, **GrdRnd2**.

Multicast size: size of multicast requests (incl. source node). We use 2 (unicast), as well as 4, 8, and 16 (true multicast requests), which we denote as **M2**, **M4**, **M8**, and **M16**, respectively.

Workload class: location of sources and targets in multicast requests, chosen according to the **uniform** and **in-line** model.

7.3 Results

We now report the most important results of the simulation analysis. We first focus on the performance of the underlying geographic routing layer. We found that in all our experiments, the greedy algorithm yields the best results in terms of congestion. As

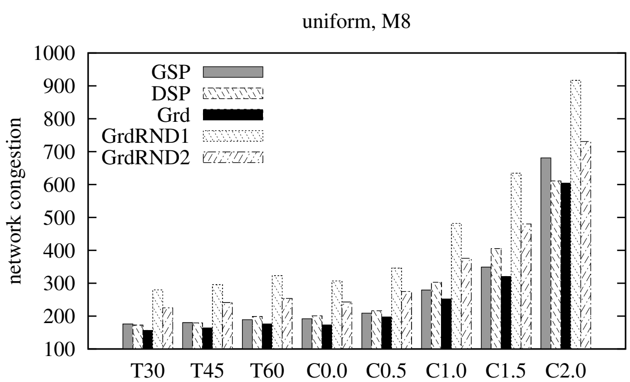

Figure 4: Comparison of Geographic Routing Algorithms

an example, Figure 4 shows the network congestion incurred by the various geographic routing primitives under a workload of uniform multicast requests of size 8, in combination with every variant of our scheme. In these scenarios, the greedy algorithm (**Grd**) is always the one that causes the lowest congestion, and as it turns out, all other scenarios show similar results. This result is particularly interesting and positive because **Grd** is also the simplest geographic algorithm available. Therefore, we dismiss all other underlying routing algorithms for the rest of our analysis.

The next question we consider is how the network congestion is affected by the selection of the intermediate point. The histogram of Figure 4 already indicates that the angle-based selection methods **T30**, **T45**, and **T60** work better than the method based on the circular arc for distance factor $\gamma > 1$.

Figure 5 confirms this result. The two graphs show the network congestion incurred by the various selection methods as a function of the size of the multicast requests, for **uniform** and **in-line** workloads, respectively. The conclusion we can draw from these experiments is that the angle-based schemes achieve the best results, only slightly better than the circular-arc method with radius less than the distance ($\gamma < 1$), and that the circular-arc method shows definitely

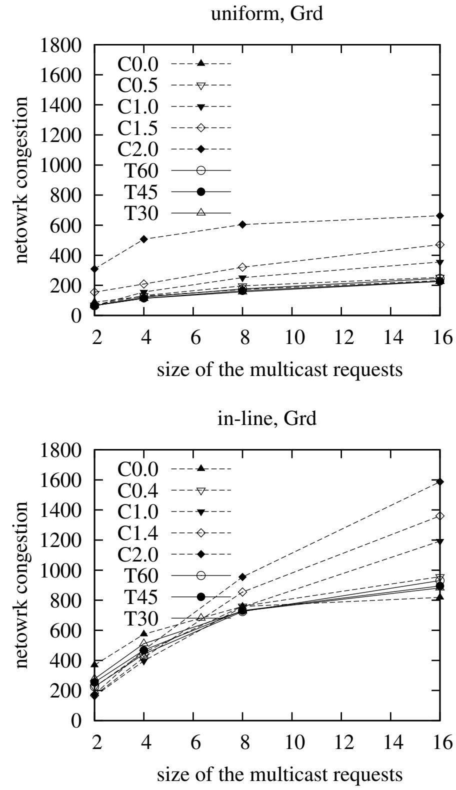

Figure 5: Comparison of Intermediate-Point Selection Methods

worse performance with higher radii ($\gamma > 1$) with proportionally worse outcomes in the case of uniform workloads. Also note that for the **in-line** model, the deterministic **C0** algorithm is the worst one for small multicast requests and it becomes the best algorithm for large multicast requests. For small requests, when routing deterministically, in the **in-line** scenario many routing paths overlap and by routing around, the congestion can be reduced. For large requests with all destinations on a line, the message has to be sent along the whole line anyway, so that sending it directly along the line becomes cheaper.

The graphs of Figure 5 also demonstrate that our multicast routing scheme performs well in an absolute sense and in particular they seem to indicate that the scheme scales gracefully with a sub-linear relation between the size of the multicast requests and congestion. Recall that all workloads consist of 1000 requests, so, for example, in the case of requests of size 16, that means that each of the 1000 messages must be delivered to 15 destinations. Consider this scenario in the extreme case of requests in which all destinations lay on a line (or a narrow band) in the network, which corresponds to the case of the **in-line** workloads. It is interesting to notice that in this case, the scheme is capable of routing all requests in such a way that the maximally-loaded node sees the equivalent of a worst-case set of *unicast* requests.

8. REFERENCES

[1] I. Abraham, D. Dolev, and D. Malkhi. LLS: A locality aware location service for mobile ad hoc networks. In *Proc. 2nd Workshop on Foundations of Mobile Comp. (DIALM-POMC)*, pages 75–84, 2004.

[2] I. Abraham, C. Gavoille, and D. Ratajczak. Compact multicast routing. In *Proc. of 23rd Symp. on Distributed Computing (DISC)*, pages 364–378, 2009.

[3] S. Arora. Polynomial time approximation scheme for Euclidean TSP and other geometric problems. In *Proc. 37th Symp. on Found. of Comp. Sc. (FOCS)*, pages 2–11, 1996.

[4] L. Barrière, P. Fraigniaud, L. Narayanan, and J. Opatrny. Robust position-based routing in wireless ad hoc networks with irregular transmission ranges. *Wireless Communication and Mobile Computing*, 3:141–153, 2003.

[5] Y. Bartal and S. Leonardi. On-line routing in all-optical networks. *Theor. Comp. Sc.*, 221(1-2):19–39, 1999.

[6] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. In *Proc. Discrete Algorithms and Methods for Mobility (DIALM)*, pages 48–55, 1999.

[7] J. Broch, D. Maltz, D. Johnson, Y.-C. Hu, and J.Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Mobile Computing and Networking*, pages 85–97, 1998.

[8] C. Busch, M. Magdon-Ismail, and J. Xi. Oblivious routing on geometric networks. In *Proc. 17th Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 316–324, 2005.

[9] C. Busch, M. Magdon-Ismail, and J. Xi. Optimal oblivious path selection on the mesh. In *Proc. 19th Int. Parallel and Distributed Processing Symp. (IPDPS)*, 2005.

[10] R. Flury and R. Wattenhofer. MLS: An efficient location service for mobile ad hoc networks. In *Proc. 7th Symp. on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, pages 226–237, 2006.

[11] J. Gao and L. Zhang. Trade-offs between stretch factor and load-balancing ratio in routing on growth-restricted graphs. *IEEE Trans. on Parallel and Distributed Systems*, 20(2):171–179, 2009.

[12] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, volume 353, chapter 5. 1996.

[13] B. Karp and H. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Proc. 6th Int. Conf. on Mobile Computing and Networking (MOBICOM)*, pages 243–254, 2000.

[14] P. Kolman and C. Scheideler. Improved bounds for the unsplittable flow problem. *J. of Algorithms*, 61(1):20–44, 2006.

[15] E. Kranakis, H. Singh, and J. Urrutia. Compass routing on geometric networks. In *Proc. 11th Canadian Conference on Computational Geometry*, pages 51–54, 1999.

[16] F. Kuhn, R. Wattenhofer, and A. Zollinger. Ad-hoc networks beyond unit disk graphs. *Wireless Networks*, 14(5):715–729, 2008.

[17] F. Kuhn, R. Wattenhofer, and A. Zollinger. An algorithmic approach to geographic routing in ad hoc and sensor networks. *IEEE/ACM Transactions on Networking*, 16:51–62, 2008.

[18] F. Li and Y. Wang. Circular sailing routing for wireless networks. In *Proc. 27th Int. Conf. on Computer Communications (INFOCOM)*, pages 1346–1354, 2008.

[19] J. Li, J. Jannotti, D. De Couto, D. Karger, and R. Morris. A scalable location service for geographic ad-hoc routing. In *Proc. 6th Int. Conf. on Mobile Comp. and Networking (MOBICOM)*, pages 120–130, 2000.

[20] B. Maggs, F. Meyer auf der Heide, B. Vöcking, and M. Westermann. Exploiting locality for networks of limited bandwidth. In *Proc. 38th Symp. on Foundations of Comp. Science (FOCS)*, pages 284–293, 1997.

[21] C. Maihofer. A survey of geocast routing protocols. *Communications Surveys & Tutorials*, 6(2):32–42, 2004.

[22] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance vector routing. In *Proc. of 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1999.

[23] L. Popa, A. Rostamizadeh, R. M. Karp, C. H. Papadimitriou, and I. Stoica. Balancing traffic load in wireless networks with curveball routing. In *Proc. 8th Symp. on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, pages 170–179, 2007.

[24] H. Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proc. 40th Symp. on Theory of Computing (STOC)*, pages 255–263, 2008.

[25] H. Räcke. Survey on oblivious routing strategies. In *Proc. 5th Conf. on Computability in Europe (CiE)*, pages 419–429, 2009.

[26] E. Royer and C. Toh. A review of current routing protocols for ad-hoc mobile wireless networks. In *IEEE Personal Communications*, volume 6, April 1999.

[27] E. M. Royer and C. E. Perkins. Multicast operation of the ad hoc on-demand distance vector routing protocol. In *Proc. 5th Int. Conf. on Mobile Comp. and Networking (MOBICOM)*, pages 207–218, 1999.

[28] H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Transactions on Communications*, 32(3):246–257, 1984.

[29] L. G. Valiant and G. J. Brebner. Universal schemes for parallel communication. In *Proc. of 13th Symp. on Theory of computing (STOC)*, pages 263–277, 1981.

[30] U. Varshney. Multicast over wireless networks. *Communications of the ACM*, 45(12):31–37, 2002.

[31] J. Xie, R. R. Talpade, A. Mcauley, and M. Liu. AMRoute: ad hoc multicast routing protocol. *Mobile Networks and Applications*, 7(6):429–439, 2002.

[32] X. Yu, X. Ban, W. Zeng, R. Sarkar, X. Gu, and J. Gao. Spherical representation and polyhedron routing for load balancing in wireless sensor networks. In *Proc. 30th Int. Conf. on Computer Communications (INFOCOM)*, pages 621–625, 2011.