Chapter 13
Oblivious Routing for Sensor Network Topologies

Costas Busch and Malik Magdon-Ismail and Jing Xi

Abstract We present oblivious routing algorithms whose routing paths are constructed independent of each other, with no dependence on the routing history. Oblivious algorithms are inherently adaptive to dynamic packet traffic, exhibit low congestion, and require low maintenance. All these attributes make oblivious algorithms to be suitable for sensor networks which are characterized by their limited energy and computational resources. Specifically, low congestion provides load balancing, and low stretch provides low energy utilization. We present two simple oblivious routing algorithms. The first algorithm is for *geometric networks* in which nodes are embedded in the Euclidean plane. In this algorithm, a packet path is constructed by first choosing a random intermediate node in the space between the source and destination, and then the packet is sent to its destination through the intermediate node. In the second algorithm we study *mesh networks*, where the nodes are arranged in a 2-dimensional grid. Grids are interesting symmetric topologies which can be used as a testbed for designing efficient new routing algorithms in sensor networks. The oblivious algorithm in the mesh constructs the paths by decomposing the network into smaller submeshes in a hierarchical manner. This algorithm can be extended to d-dimensions, which makes it suitable for three-dimensional sensor network deployments, such as in buildings and tall structures. We analyze the algorithms in terms of the stretch and congestion of the resulting paths, and demonstrate that they exhibit near optimal performance.

Busch Costas
Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803, USA e-mail: busch@csc.lsu.edu

Magdon-Ismail Malik
Department of Computer Science Dept, Rensselaer Polytechnic Institute, Troy, NY 12180, USA e-mail: magdon@cs.rpi.edu

Xi Jing
Department of Computer Science Dept, Rensselaer Polytechnic Institute, Troy, NY 12180, USA e-mail: xij2@cs.rpi.edu (Currently employed at Bank of America, New York)

13.1 Introduction

Routing algorithms specify the paths to be followed by packets in a network. A routing algorithm is *oblivious* if the path of every packet is given independently of the paths of the other packets and without considering the history of the previously routed packets. Oblivious algorithms are by their nature distributed and capable of solving online routing problems, where packets continuously arrive in the network. Hence, oblivious routing is often preferred to non-oblivious routing, since one does not need to make assumptions regarding the nature of the traffic.

In wireless and sensor network applications the nodes may have power and computing capability constraints. For example, a sensor node is typically operated with a battery that has limited energy capacity. To maximize the lifetime of the nodes, the time until nodes run out of power, it is important to minimize the utilization of individual nodes. This relates directly to balancing the packet traffic for minimizing the node congestion. With load balancing, the lifetime of a battery operated sensor network is prolonged, since the time that the first node runs out of energy is extended. Further, using paths of small stretch (ratio of path length to shortest path) is also beneficial to the sensor network since small stretch implies low overall energy utilization.

Oblivious routing algorithms are suitable for balancing the congestion in the network and therefore extending the lifetime of the nodes. They can also provide small stretch in interesting sensor network deployment scenarios. Oblivious algorithms are easy to implement in wireless and sensor networks, on account of their simplicity. We present two oblivious routing algorithms which are suitable for wireless sensor networks. The first algorithm is for *geometric networks*, while the second is for *mesh networks*. Both of these algorithms achieve low congestion and stretch. We continue with describing each of them.

13.1.1 Geometric Networks

We present the oblivious routing algorithm for geometric networks which was originally proposed in [7]. In geometric networks, the nodes are placed in the 2-dimensional Euclidian space and we assume that all the nodes are contained in some geographic area $\mathscr{A}$ (Figure 13.1). Suppose that a packet wants to go from a node s to a node t in the network. The algorithm is to choose a random intermediate node w in the space between the s and t, then sends the packet to w, and then sends the packet from w to its destination (see Figure 13.3). In order to implement this idea, we assume that between every pair of nodes there is a dedicated path which we call the *default path*. For example, the default path between two nodes u and v could be a shortest path that connects them. We denote the set of all default paths by Q. The choice of the default paths affects the performance of our algorithm, and the closer the default paths are to the geodesics, the lines that connect the respective end points of the paths, the better the performance of the algorithm.

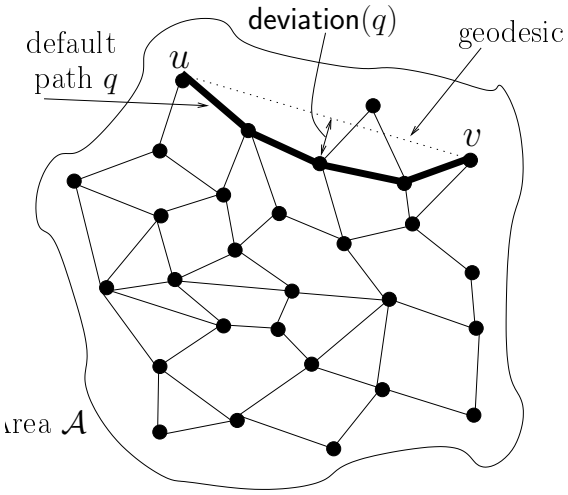

Fig. 13.1 Example of a geometric network

We analyze the algorithm in terms of stretch and congestion. Consider some set of paths P produced by the routing algorithm. Denote by $\mathsf{stretch}(P)$ the maximum ratio of a path length to the length of the respective shortest path (the length is measured in the number of node hops). The *node congestion* C_{node} is the maximum number of paths that use any node in the network. The *edge congestion* C_{edge} is the maximum number of paths that use any edge in the network. Let C^*_{node} and C^*_{edge} denote the optimal node and edge congestions, which could be obtained by a brute force search through all possible paths from the sources to the destinations in P.

The stretch and congestion of the paths P produced by our algorithm depend on the quality of the default paths Q. In particular, provided that the geometric embedding is "faithful" to the topology of the network (i.e. nodes far apart are connected with more hops than nodes closer to each other) we obtain:

$$\begin{aligned}\mathsf{stretch}(P) &= O\big(\mathsf{stretch}(Q)\big),\\ C_{node} &= O\big(C^*_{node}\cdot(1+\mathsf{deviation}^3(Q))\cdot\log(n+\mathsf{deviation}(Q))\big),\end{aligned}$$

where n is the number of nodes, and $\mathsf{deviation}(Q)$ measures the extent of deviation of the default paths from geodesics (see Figure 13.1). We also obtain a corresponding result for the edge congestion. The congestion results hold with high probability, while the stretch result is deterministic.

We apply our general result to two particular geometric networks: the 2-dimensional mesh network, and uniformly distributed disc networks. Both of these kinds of network have geometric embeddings that are faithful to the network topologies. The mesh network is a 2-dimensional grid of nodes (see Figure 13.2). In disc networks, each node is connected to any node within a specific disc radius. In the uniformly

distributed disc graphs, each unit square area contains a constant number of nodes. In these networks, we can choose default paths with constant stretch and deviation. Therefore, our algorithm gives paths with *constant* stretch. We obtain node and edge congestions which are within logarithmic factors of optimal, $C_{node} = O(C^*_{node} \cdot \log n)$ and $C_{edge} = O(C^*_{edge} \cdot \log n)$, with high probability. Maggs *et al.* [21] give a worst case edge congestion lower bound of $\Omega(C^*_{edge} \cdot \log n)$ for any oblivious routing algorithm in the 2-dimensional mesh. Therefore, in addition to constant stretch, the congestion we obtain is optimal, within constant factors, for oblivious algorithms.

13.1.2 Mesh Networks

We continue with an alternative oblivious algorithm for mesh networks. A 2-dimensional mesh with n nodes is simply a $\sqrt{n} \times \sqrt{n}$ grid of nodes (see Figure 13.2). The oblivious algorithm for geometric networks that we described above can be applied to the 2-dimensional mesh. However, it cannot be extended to higher dimensions. Here, we present the oblivious routing algorithm in the mesh which originally appears in [8]. The algorithm can be used for any d-dimensional mesh network ($d \geq 2$) and it provides near optimal congestion while maintaining a small stretch. (For general networks this is not feasible, as is explained in Section **??**.)

The benefit of a higher dimensional algorithm is that it can be used in sensor network deployments in buildings and other tall structures which are not restricted in the 2-dimensional space. Note that the mesh topology has a symmetric topology and it may not be directly applicable in real deployment scenarios. However, the mesh topology exhibits many characteristics which can be found in real sensor networks (and especially in random uniform area deployment), such as for example the low node degree, small ratio of Euclidian to graph distance, and low doubling dimension. Routing algorithms on the mesh often generalize to other network topologies as well and typically the mesh is used as an exploration testbed for the design and analysis of new efficient algorithms.

Given a routing problem (collection of sources and destinations), let $C^* = C^*_{edge}$ denote the optimal edge congestion attainable by any routing algorithm (oblivious or not). We give an oblivious routing algorithm for the d-dimensional mesh with n nodes, that achieves congestion $O(d \cdot C^* \cdot \log n)$, and stretch $O(d^2)$, For the d-dimensional mesh with n nodes, Maggs et al. [21] give the lower bound $C^*_{obl} = \Omega\left(\frac{C^*}{d} \cdot \log n\right)$ in the worst case for any oblivious routing algorithm. Considering the class of oblivious algorithms, our algorithm is within $O(d^2)$ of optimal for both congestion and dilation. For a fixed d, our algorithm is optimal to within constant factors.

Our algorithm is based upon a hierarchical decomposition of the mesh. Starting at its source node, a packet constructs its path by randomly selecting intermediate points in submeshes of increasing size until the current submesh contains the destination node. Random intermediate points are then selected in submeshes of decreas-

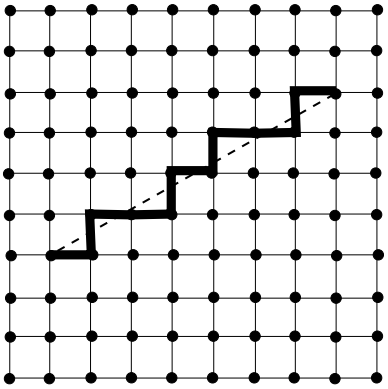

Fig. 13.2 The 2-dimensional mesh network and a shortest path between two nodes

ing size until the destination node is reached. The key new idea we introduce is the notion of "bridge" submeshes that make it possible to move from a source to a destination more quickly, without increasing the congestion. These bridge submeshes are instrumental in controlling the stretch, while maintaining low congestion.

13.2 Geometric Networks

13.2.1 Preliminaries on Geometric Networks

Consider a geometric network G with n nodes which is embedded in the Euclidean plane, $\mathscr{R}^2$ (see Figure 13.1). We assume that G is un-weighted, undirected, connected and stationary. Further, its edges are un-weighted, i.e. the communication cost of every link is 1 regardless of the link's Euclidian distance. Every node v_i has a position $\mathbf{x}_i \in \mathscr{R}^2$. We will also use the notation $\mathbf{x}(v)$ to denote the position of the node v. The network is defined over some *area* $\mathscr{A}$. We will also refer to the network itself as $\mathscr{A}$ when the context is clear. Thus, $\mathbf{x}_i \in \mathscr{A}$ for all i. For the area $\mathscr{A}$, we define a *coverage radius* $R(\mathscr{A})$ of the area as follows (we drop the $\mathscr{A}$ dependence when the context is clear). If, for every point $\mathbf{x} \in \mathscr{A}$, there is at least one node v that is located at most a (Euclidean) distance R from $\mathbf{x}$, then R is a coverage radius, i.e., from any point in $\mathscr{A}$, one needs to go a distance of at most R to reach some node in the network.

We define the *pseudo-convexity* $\gamma(\mathscr{A})$ of area $\mathscr{A}$ as follows. Let $\mathbf{x}_1, \mathbf{x}_2 \in \mathscr{A}$, and consider the line ℓ joining $\mathbf{x}_1$ to $\mathbf{x}_2$. Let $\ell^\perp$ be a line of equal length to ℓ such that ℓ and $\ell^\perp$ are mutually perpendicular bisectors. Let $\ell^\perp_{\mathscr{A}} \subseteq \ell^\perp$ be the intersection of $\ell^\perp$ with $\mathscr{A}$. Denote by $|\ell^\perp_{\mathscr{A}}|$ the measure, or "length" of $\ell^\perp_{\mathscr{A}}$. We define the local pseudo-convexity at $\mathbf{x}_1, \mathbf{x}_2$ as $\gamma(\mathbf{x}_1, \mathbf{x}_2) = |\ell^\perp_{\mathscr{A}}|/|\ell^\perp|$. The pseudo-convexity γ of $\mathscr{A}$

is the infimum over all pairs $\mathbf{x}_1, \mathbf{x}_2 \in \mathscr{A}$ of $\gamma(\mathbf{x}_1, \mathbf{x}_1)$,

$$\gamma = \infty_{\mathbf{x}_1, \mathbf{x}_2 \in \mathscr{A}} \gamma(\mathbf{x}_1, \mathbf{x}_2).$$

In words, γ is a lower bound on the fraction of the perpendicular bisector $\ell^\perp$ that is guaranteed to be in $\mathscr{A}$. Note that $\mathscr{A}$ is convex if $\gamma \geq \frac{1}{2}$ but that the converse is not true (consider a very thin rectangle). For any regular convex polygon, or a circle, $\gamma \geq \frac{1}{2}$. For a network embedded in a fixed area $\mathscr{A}$, γ is independent of n, which will have important consequences on the optimality of our path selection algorithm (provided that $\gamma > 0$).

Since the network is embedded in $\mathscr{R}^2$, there are two notions of distance between two nodes u, v that are useful. The first is the *Euclidean distance*, $\mathsf{dist}_E(u,v)$ which is the length of the straight line (or *geodesic*) joining the positions $\mathbf{x}(u)$ and $\mathbf{x}(v)$. For two points $\mathbf{x}, \mathbf{y} \in \mathscr{R}^2$, $\|\mathbf{x} - \mathbf{y}\|$ is the Euclidean distance between them. Thus, $\mathsf{dist}_E(u,v) = \|\mathbf{x}(u) - \mathbf{x}(v)\|$. The second useful distance measure is the graph-theoretic or *network distance* $\mathsf{dist}_G(u,v)$ which is the length of the shortest path in G from u to v. For any path p in G, we use $|p|$ to denote the length of the path (number of edges in the path), and we define the *Euclidean path length* $|p|_E$ to be the weighted path length, where the weights on the edges are set to the Euclidean distance between the nodes they connect.

For two nodes u, v, we use the measure $\mathsf{dist}_G(u,v)/\mathsf{dist}_E(u,v)$ to represent how well the Euclidean distances in the network embedding represent the network distances. We introduce two parameters α, β to denote lower and upper bounds for this measure. Thus, for every pair of nodes u, v,

$$\alpha \leq \frac{\mathsf{dist}_G(u,v)}{\mathsf{dist}_E(u,v)} \leq \beta$$

Thus, two nodes u, v that are connected by an edge ($\mathsf{dist}_G(u,v) = 1$) cannot be separated by more than a distance of $\frac{1}{\alpha}$. Note also that $\mathsf{dist}_G(u,v) \geq 1$, so $\mathsf{dist}_E(u,v) \geq \frac{1}{\beta}$. We thus have the following useful lemma,

Lemma 1. *For any two nodes, u, v, $\mathsf{dist}_E(u,v) \geq \frac{1}{\beta}$. If u and v are adjacent then $\mathsf{dist}_E(u,v) \leq \frac{1}{\alpha}$.*

Lemma 1 allows us to derive an upper bound on the number of nodes that can be in a disc.

Lemma 2. *Consider a disc of radius $r \geq \frac{1}{\beta}$ containing M nodes. Then $M \leq c(\beta r)^2$, where c is a constant, $c \leq 1 + \pi/(\frac{2\pi}{3} - \frac{\sqrt{3}}{2})$.*

Proof: The intuition is that every node accounts for an area of at least π/β^2. Since the total area is πr^2, there can be at most $\pi r^2/(\pi/\beta^2) = (\beta r)^2$ nodes. The only complication is that nodes near the boundary do not take up the entire area π/β^2, as part of this area could be outside the disc. Taking this boundary phenomenon into account gives us the constant c.

To prove the lemma, consider the circle of radius $r-\frac{1}{\beta}$ with M_1 nodes, and the remaining ring from $r-\frac{1}{\beta}$ to r with M_2 nodes. Since every one of the M_1 nodes defines an area of radius $\frac{1}{\beta}$ that is completely enclosed in the disc, we have the $M_1 \leq (\beta r)^2$. Now consider the ring. The smallest area blocked off by a node occurs when the node is on the boundary, in which case the area is smallest when $r = \frac{1}{\beta}$. Some geometric considerations show that this area blocked of is at least $\frac{1}{\beta^2}(\frac{2\pi}{3} - \frac{\sqrt{3}}{2})$, and since the area of the ring is at most πr^2, $M_2 \leq (\beta r)^2 \pi / (\frac{2\pi}{3} - \frac{\sqrt{3}}{2})$. To conclude, note that $M \leq M_1 + M_2$. ⊠

For every pair of nodes u, v, we assume that a *default path* $q(u,v)$ in G is provided. For example, the default paths could be the shortest paths connecting the pairs of nodes. The default paths should actually have certain good properties and may not be shortest paths. Denote the set of all $n(n-1)$ default paths by the set Q. For a given default path $q(u,v)$, we define the stretch of the path, $\mathsf{stretch}(q)$, to be $|q(u,v)|/\mathsf{dist}_G(u,v)$ which is the factor by which q is longer than the shortest path between u and v.

Consider the infinite line ℓ drawn through the points $\mathbf{x}(u)$ and $\mathbf{x}(v)$. Let z be any intermediate node in the path $q(u,v)$. The *displacement* of z from ℓ is the perpendicular (Euclidean) distance from $\mathbf{x}(z)$ to ℓ. The deviation of $q(u,v)$ from ℓ, denoted $\mathsf{deviation}(q)$, is the maximum displacement of any intermediate node z of q from ℓ. In other words, $\mathsf{deviation}(q)$ measures how closely the path $q(u,v)$ stays to the straight line (geodesic) from $\mathbf{x}(u)$ to $\mathbf{x}(v)$.

The stretch factor for the entire set of paths Q is the maximum stretch of any path in Q, and similarly with the deviation of Q. We use Σ to denote the stretch and Δ to denote the deviation,

$$\Sigma_Q = \mathsf{stretch}(Q) = \max_{q \in Q} \mathsf{stretch}(q),$$
$$\Delta_Q = \mathsf{deviation}(Q) = \max_{q \in Q} \mathsf{deviation}(q).$$

As we will see later in the analysis of our path selection algorithm, if the default paths have small stretch and deviation, then the path selection performance is closer to optimal. Thus, it is beneficial to select default paths that make these parameters as small as possible. We will see later that for a variety of networks they can be made constants.

13.2.2 Oblivious Routing on Geometric Networks

Here we describe our oblivious routing algorithm. The task of the algorithm is to provide a path for each packet in the network. It is assumed that each node knows the default paths that connect it to other nodes in the network. The algorithm is randomized and we assume that each node has access to a sequence of random

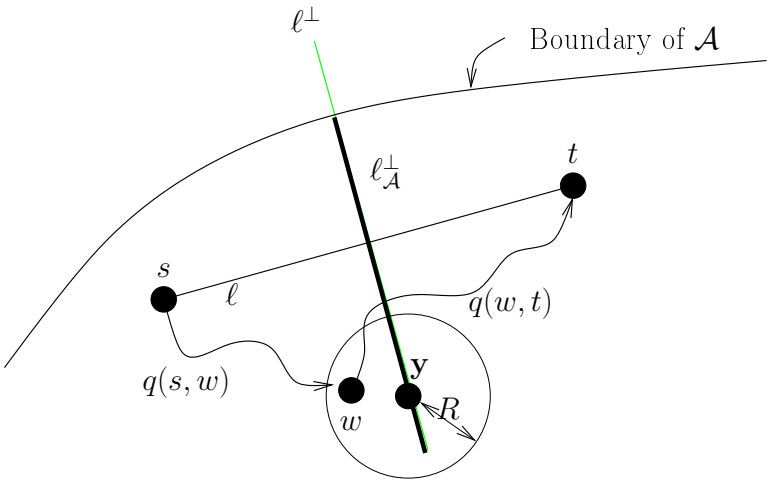

Fig. 13.3 Path selection with oblivious routing algorithm for geometric networks

Algorithm 11 Oblivious Routing for Geometric Networks

Input: A graph G embedded in an area $\mathscr{A}$ with default paths Q; and a packet π with source s and destination t;

Output: A path $p(s,t)$ from s to t;

1: Let ℓ be the geodesic line segment that connects $\mathbf{x}(s)$ and $\mathbf{x}(t)$. Let $\ell^\perp$ be the perpendicular bisector of ℓ which has the same length as ℓ and is also bisected by ℓ. Let $\ell^\perp_{\mathscr{A}}$ be the part of $\ell^\perp$ inside $\mathscr{A}$;
2: Choose a point $\mathbf{y}$ randomly and uniformly on $\ell^\perp_{\mathscr{A}}$;
3: Find a node w close to $\mathbf{y}$ within coverage radius R;
4: The path $p(s,t)$ from s to t is formed by concatenating the default paths $q(s,w)$ and $q(w,t)$:

$$p(s,t) = q(s,w)q(w,t);$$

numbers. The path selection algorithm is executed for each packet independently of every other packet, so the algorithm is oblivious, and thus distributed and online. Algorithm 11 is the detailed algorithm for a particular packet. The algorithm is similar for any other packet. Figure 13.3 graphically illustrates the algorithm.

In the analysis of the algorithm, we consider a set of N packets Π which we will refer to with their sources and destinations, $\Pi = \{s_i, t_i\}_{i=1}^N$. The result of applying the algorithm to each packet is a set of paths $P = \{p_i\}_{i=1}^N$, where each path $p_i \in P$ is from the source node s_i to the destination node t_i. We define the stretch for a path $p \in P$ as well as the stretch factor for the entire set P as we did in Section 13.2.1 with the default paths Q. We define D^* as the maximum shortest path length between any pair of sources and destinations in Π, namely, $D^* = \max_i \mathsf{dist}_G(s_i, t_i)$. We first

analyze the stretch of paths P and then we continue with the node-congestion and edge-congestion.

13.2.2.1 Stretch Analysis

We now give a bound on $\mathsf{stretch}(P)$, the stretch factor of the paths selected.

Theorem 1. $\mathsf{stretch}(P) \leq \frac{\sqrt{2}\beta}{\alpha} \cdot \Sigma_Q \cdot (1+\sqrt{2}R\alpha)$.

Proof: We will refer to Figure 13.3 in our proof. By construction, $\sqrt{2}\|\mathbf{x}(s)-\mathbf{y}\| \leq \|\mathbf{x}(s)-\mathbf{x}(t)\|$, and $\|\mathbf{x}(s)-\mathbf{y}\| = \|\mathbf{x}(t)-\mathbf{y}\|$. Since $\|\mathbf{x}(w)-\mathbf{y}\| \leq R$, by the triangle inequality, we have that

$$\begin{aligned}\|\mathbf{x}(s)-\mathbf{x}(w)\| &\leq \|\mathbf{x}(s)-\mathbf{y}\| + \|\mathbf{x}(w)-\mathbf{y}\| \\ &\leq \tfrac{1}{\sqrt{2}}\|\mathbf{x}(s)-\mathbf{x}(t)\| + R.\end{aligned}$$

Similarily,

$$\|\mathbf{x}(t)-\mathbf{x}(w)\| \leq \tfrac{1}{\sqrt{2}}\|\mathbf{x}(s)-\mathbf{x}(t)\| + R.$$

From the definition of Σ_Q, the stretch factor of the default paths, we have:

$$\begin{aligned}|q(s,w)| &\leq \Sigma_Q \cdot \mathsf{dist}_G(s,w) \\ &\leq \beta \cdot \Sigma_Q \cdot \mathsf{dist}_E(s,w),\end{aligned}$$

and similarily

$$|q(w,t)| \leq \beta \cdot \Sigma_Q \cdot \mathsf{dist}_E(w,t).$$

We thus conclude that

$$\begin{aligned}|p(s,t)| &= |q(s,w)| + |q(w,t)|, \\ &\leq \beta \cdot \Sigma_Q \cdot (\|\mathbf{x}(s)-\mathbf{x}(w)\| + \|\mathbf{x}(t)-\mathbf{x}(w)\|), \\ &\leq \beta \cdot \Sigma_Q \cdot (\sqrt{2}\|\mathbf{x}(s)-\mathbf{x}(t)\| + 2R).\end{aligned}$$

Since $\|\mathbf{x}(s)-\mathbf{x}(t)\| \leq \frac{1}{\alpha}\mathsf{dist}_G(s,t)$, and $\mathsf{dist}_G(s,t) \geq 1$, we obtain the theorem. ⊠

Typically R, α, β are constants, in which case $\mathsf{stretch}(P) = O(\mathsf{stretch}(Q))$, i.e., the stretch factor of the algorithm is determined by the quality of the default paths.

13.2.2.2 Node Congestion Analysis

We now turn to the node congestion. We will get a bound on the expected congestion for any particular node with respect to the optimal congestion. We will then use a Chernoff bounding argument to obtain a high probability result.

To bound the expected node congestion for a particular node v, we need to understand the probability that a particular packet might use the node. Thus consider

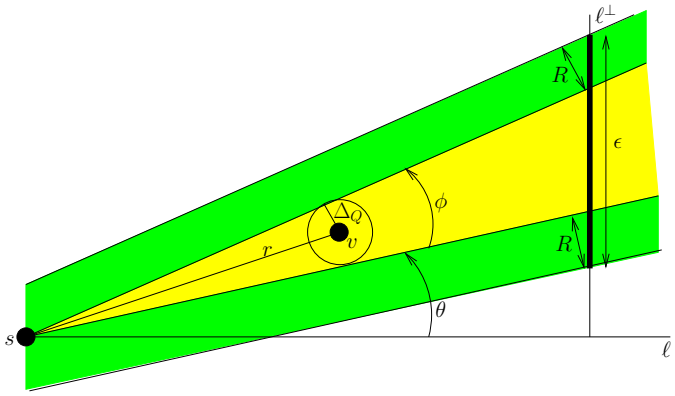

Fig. 13.4 Probability of using a node v

a particular packet π, with source s and destination t, which uses intermediate node w. Phase I of the path $p(s,t)$ corresponds to the first part $q(s,w)$, while phase II to the second part $q(w,t)$. Suppose that the packet uses v in phase I of its path (we will bound the probability that π uses v in phase I of its path, a similar argument applies to phase II of the path). Let r denote $\|\mathbf{x}(v)-\mathbf{x}(s)\|$. The situation is illustrated in Figure 13.4. A circle of radius Δ_Q is drawn around v. We give an upper bound on the probability that π uses node v in the following lemma,

Lemma 3. *Suppose that packet π has source s and destination t. Let P_I be the probability that π uses node v in phase I of its path and P_{II} be the probability that π uses node v in phase II of its path. Then,*

$$P_I \leq \frac{5}{\gamma}\left(\frac{R}{\|\mathbf{x}(s)-\mathbf{x}(t)\|}+\frac{\Delta_Q}{\|\mathbf{x}(s)-\mathbf{x}(v)\|}\right),$$
$$P_{II} \leq \frac{5}{\gamma}\left(\frac{R}{\|\mathbf{x}(s)-\mathbf{x}(t)\|}+\frac{\Delta_Q}{\|\mathbf{x}(t)-\mathbf{x}(v)\|}\right).$$

Proof: Consider the shaded cone subtended by the source s, tangent to the circle of radius Δ_Q centered on v. Since the deviation of the default paths is Δ_Q, The intermediate node must lie within the shaded cone if the path $q(s,w)$ is to pass through v. If the intermediate node is in the cone, the random intermediate point $\mathbf{y}$ must lie either in the cone or in one of the two shaded strips of thickness R around the cone. Since $\mathbf{y}$ must also be on $\ell^\perp$, $\mathbf{y}$ must lie on the line segment illustrated by the thick line of length ε illustrated in Figure 13.4. The probability of using v is then bounded by $\varepsilon/|\ell^\perp_{\mathscr{A}}|$. We use the definitions of θ,ϕ as shown in Figure 13.4. Using some elementary geometry, we find that

$$\varepsilon = R\cdot\left(\frac{1}{\cos(\theta)}+\frac{1}{\cos(\theta+\phi)}\right)+\frac{1}{2}|\ell|\cdot(\tan(\theta+\phi)-tan(\theta)).$$

We observe that ε is largest when $\theta \leq \frac{\pi}{4}$ and $\theta + \phi \leq \frac{\pi}{4}$, so using some trigonometric identities we get:

$$\begin{aligned}\varepsilon &\leq 2\sqrt{2}R + \frac{|\ell|}{2} \cdot \frac{\tan\phi(1+\tan^2\theta)}{1-\tan\theta\tan\phi}, \\ &\leq 2\sqrt{2}R + |\ell| \cdot \frac{\tan\phi}{1-\tan\phi},\end{aligned}$$

where the last line follows because $\tan\theta < 1$. Since $|\ell^{\perp}_{\mathscr{A}}| \geq \gamma|\ell^{\perp}| = \gamma|\ell|$, we get that the probability of using v is at most

$$\begin{aligned}\text{Prob} &\leq 2\sqrt{2}\frac{R}{\gamma|\ell|} + \frac{1}{\gamma}\frac{\tan\phi}{1-\tan\phi}, \\ &\overset{(a)}{\leq} 2\sqrt{2}\frac{R}{\gamma|\ell|} + \frac{2\tan\phi}{\gamma}, \\ &\overset{(b)}{=} 2\sqrt{2}\frac{R}{\gamma|\ell|} + \frac{4}{\gamma}\frac{\tan\frac{\phi}{2}}{1-\tan^2\frac{\phi}{2}}, \\ &\overset{(c)}{\leq} 2\sqrt{2}\frac{R}{\gamma|\ell|} + \frac{64\tan\frac{\phi}{2}}{15\gamma}, \\ &\overset{(d)}{=} 2\sqrt{2}\frac{R}{\gamma|\ell|} + \frac{64}{15\gamma}\frac{\Delta_Q/r}{\sqrt{1-\Delta_Q^2/r^2}}, \\ &\overset{(e)}{\leq} \frac{5}{\gamma}\left(\frac{R}{|\ell|} + \frac{\Delta_Q}{r}\right).\end{aligned}$$

Inequality (a) follows because when $\tan\phi \leq \frac{1}{2}$, $\tan\phi/(1-\tan\phi) \leq 2\tan\phi$, and when $\tan\phi > \frac{1}{2}$, $2\tan\phi > 1$, in which case it is a trivially valid upper bound for the probability. (b) follows by using a double angle identity. (c) follows by a similar argument that lead to (a) by considering separately $\tan\frac{\phi}{2} \leq \frac{1}{4}$ and $\tan\frac{\phi}{2} > \frac{1}{4}$. (d) follows because from Figure 13.4, we see that $\tan\frac{\phi}{2} = \Delta_Q/\sqrt{r^2-\Delta_Q^2}$. Finally, (e) follows using $2\sqrt{2} < 5$ and by considering separately the cases $\Delta_Q/r \leq \frac{1}{5}$ and $\Delta_Q/r > \frac{1}{5}$ (similar with (a) and (c)).

To conclude, note that by symmetry, the situation is exactly reversed if the packet uses v in phase II of its path, except that now r will be the distance from v to the destination t. ⊠

In order to bound the congestion on node v, we need to bound the number of packets that can cross v, and then using Lemma 3 we will be able to bound the expected congestion on v. We first compute how far the packets that cross v have their sources or destinations from v, this will help to bound the number of those packets. Let X^I denote the packets that could possibly use v during phase I of their path, and similarly X^{II}. Consider only the packets in X^I. Let $S^I = \{s_k\}$ denote the

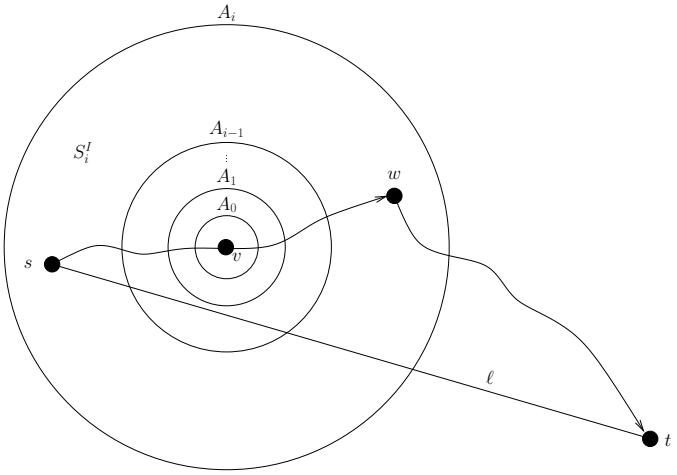

Fig. 13.5 Expected congestion at a node v

sources of all the packets in X^I. Let r_{max} be the maximum (Euclidean) distance from the positions of these sources to $\mathbf{x}(v)$, thus, $r_{max} = \max_{s \in S^I} \| \mathbf{x}(s) - \mathbf{x}(v) \|$. We have the following result (a similar result holds for the destinations).

Lemma 4. $r_{max} \leq \frac{D^*}{\sqrt{2}\alpha} + R + \Delta_Q$.

Proof: Let s be a source that could possibly use v in phase I and let t be the corresponding destination. Let w be a possible intermediate node. Then $\| \mathbf{x}(s) - \mathbf{x}(w) \| \leq \frac{1}{\sqrt{2}} \| \mathbf{x}(s) - \mathbf{x}(t) \| + R$. Since the path cannot deviate by more than Δ_Q from the line joining $\mathbf{x}(s)$ to $\mathbf{x}(w)$, and the path passes through v, it follows that

$$\begin{aligned} \| \mathbf{x}(v) - \mathbf{x}(s) \| &\leq \| \mathbf{x}(s) - \mathbf{x}(w) \| + \Delta_Q \\ &\leq \frac{1}{\sqrt{2}} \| \mathbf{x}(s) - \mathbf{x}(t) \| + R + \Delta_Q. \end{aligned}$$

To conclude, note that $\| \mathbf{x}(s) - \mathbf{x}(t) \| \leq \frac{1}{\alpha} \mathsf{dist}_G(s,t)$ and $\mathsf{dist}_G(s,t) \leq D^*$. ⊠

In order to bound the congestion on v, we will divide the area around v into concentric rings with maximum radius r_{max}. We will then bound the number of packets that originate in each ring and use v. The number of packets from each ring will be used to bound the expected congestion caused by each ring. The sum of the expected congestions from the rings will determine the total congestion on node v.

Consider concentric rings $A_0, A_1, A_2, \ldots$ of exponentially increasing radius, centered at $\mathbf{x}(v)$. Ring A_i has radius $r_i = 2^i/\beta$, for $i \geq 0$. Let $i_{max} = \lceil \log(r_{max}\beta) \rceil$ (logarithms are base 2). Note that all the sources in S^I are contained in $A_{i_{max}}$. For $i > 0$, we collect in set S_i^I all the sources which are in ring A_i, but not in A_{i-1} (that is, they are in the area between A_{i-1} and A_i). Figure 13.5 illustrates the situation. Consider

a particular i and the packets X_i^I with sources in S_i^I. Let $N_i = |X_i^I|$ be the number of packets with sources in S_i^I. In order to obtain an upper bound on the expected congestion at v, we will need to bound N_i in terms of the optimal node congestion C^*_{node}.

Lemma 5. *For any* $i \geq 0$:

$$C^*_{node} \geq \frac{\alpha h_i N_i}{4c(\beta r_i)^2},$$

where,

$$h_i = \max\left\{\frac{1}{\beta}, \sqrt{2}(r_{i-1} - R - \Delta_Q)\right\}.$$

Proof: As in the proof of Lemma 4, $\|\mathbf{x}(s) - \mathbf{x}(v)\| \leq \frac{|\ell|}{\sqrt{2}} + R + \Delta_Q$, and since $\|\mathbf{x}(s) - \mathbf{x}(v)\| \geq r_{i-1}$, we get

$$|\ell| \geq \sqrt{2}(r_{i-1} - R - \Delta_Q).$$

From Lemma 1, $|\ell| \geq \frac{1}{\beta}$, therefore $|\ell| \geq h_i$. Furthermore, from the definition of α and Lemma 1, we have that the minimum number of hops from s to t is at least $\alpha|\ell| \geq \alpha h_i$, and each of these hops moves a distance of at most $\frac{1}{\alpha}$. So, for N_i such paths, any path selection algorithm will have to use at least αh_i hops per path, within a disc of radius

$$r = r_i + h_i \leq r_i + 2r_{i-1} \leq 2r_i.$$

By Lemma 2, there are at most $4c(\beta r_i)^2$ nodes within this disc of radius r. The minimum total number of times these nodes are used by *any* path selection algorithm is $\alpha h_i N_i$. Thus, the average number of times T_{avg} a node is used in radius r is at least

$$T_{avg} \geq \frac{\alpha h_i N_i}{4c(\beta r_i)^2},$$

where c is the constant defined in Lemma 2. Since one of these nodes has to be used at least T_{avg} times, we obtain a lower bound on the congestion for any path selection algorithm, and hence for the optimal congestion $C^*_{node} \geq T_{avg}$. ⊠

Note that inverting the bound in Lemma 5, we get an upper bound for N_i, when $i \geq 1$:

$$N_i \leq \frac{4c(\beta r_i)^2 C^*_{node}}{\alpha h_i} \tag{13.1}$$

Note that for $i = 0$ it holds trivially that $N_0 \leq 0$, since no node except for v can be in ring A_0 (a consequence of Lemma 1). The upper bound for N_i together with the upper bound for the probability that any of these packets uses node v (Lemma 3) allows us to bound the expected congestion,

Theorem 2. *The expected congestion on node v is*

$$E[C(v)] \leq f(\gamma, \alpha, \beta, R, \Delta_Q, D^*) \cdot C^*_{node},$$

where,

$$f(\gamma,\alpha,\beta,R,\Delta_Q,D^*) = \\ \frac{40c\beta^2(R+2\Delta_Q)}{\gamma\alpha}\cdot((3+4\beta(R+\Delta_Q))^2 \\ +4\log\left(\frac{\beta D^*}{\sqrt{2}\alpha}+\beta(R+\Delta_Q)\right)+1).$$

Proof: Let $Prob_v(\pi)$ be the probability that packet $\pi \in X_i^I$ uses node v. Then packet π's contribution to the expected node congestion at v is $Prob_v(\pi)$. Using Lemma 3, we can bound $Prob_v(\pi)$ by P_I. Then, N_iP_I is an upper bound for the contribution to the expected node congestion at v due to the packets in X_i^I. Since every source in S_i^I is distanced at least r_{i-1} from node v, from Lemma 3 and using (13.1), and the fact that $r_i \geq h_i$, we obtain for $i \geq 1$:

$$\begin{aligned}\sum_{\pi\in X_i^I} Prob_v(\pi) &\leq N_iP_I \\ &\leq \frac{20c(\beta r_i)^2C^*_{node}}{\gamma\alpha h_i}\left(\frac{R}{h_i}+\frac{\Delta_Q}{r_{i-1}}\right) \\ &= \frac{20c\beta^2C^*_{node}}{\gamma\alpha}\left(R\frac{r_i^2}{h_i^2}+2\Delta_Q\frac{r_i}{h_i}\right) \\ &\leq \frac{20c\beta^2(R+2\Delta_Q)C^*_{node}}{\gamma\alpha}\cdot\frac{r_i^2}{h_i^2}.\end{aligned}$$

The expected node congestion at v, is obtained by summing the contributions due to each set X_i^I for $i = 1,\ldots,i_{max}$. Thus,

$$E[C(v)] \leq \frac{20c\beta^2(R+2\Delta_Q)C^*_{node}}{\gamma\alpha}\sum_{i=1}^{i_{max}}\frac{r_i^2}{h_i^2}.$$

Consider now the ratio h_i/r_i. We have

$$\begin{aligned}\frac{h_i}{r_i} &= \frac{\max\left\{\frac{1}{\beta},\sqrt{2}(r_{i-1}-R-\Delta_Q)\right\}}{r_i} \\ &= \max\left\{\frac{1}{2^i},\sqrt{2}\left(\frac{1}{2}-\frac{\beta(R+\Delta_Q)}{2^i}\right)\right\}.\end{aligned}$$

Let

$$i^* = \left\lceil \log(\sqrt{2}+4\beta(R+\Delta_Q))\right\rceil.$$

Then for $i \geq i^*$, it holds $\frac{h_i}{r_i} \geq \frac{\sqrt{2}}{4}$, or equivalently $\frac{r_i}{h_i} \leq 2^{3/4} < 2$. For $1 \leq i < i^*$, we have that $\frac{h_i}{r_i} \geq \frac{1}{2^i}$, or in other words, $\frac{r_i}{h_i} \leq 2^i$. Since $i_{max} = \lceil \log(r_{max}\beta)\rceil$, using the bound in Lemma 4, we get:

$$\begin{aligned}\sum_{i=1}^{i_{max}} \frac{r_i^2}{h_i^2} &= \sum_{i=1}^{i^*-1} \frac{r_i^2}{h_i^2} + \sum_{i=i^*}^{i_{max}} \frac{r_i^2}{h_i^2} \\ &\leq \sum_{i=1}^{i^*-1} 4^i + \sum_{i=i^*}^{i_{max}} 4 \\ &\leq (2^{i^*})^2 + 4i_{max}, \\ &\leq (3+4\beta(R+\Delta_Q))^2 \\ &\quad +4\log\left(\frac{\beta D^*}{\sqrt{2}\alpha} + \beta(R+\Delta_Q)\right) + 1.\end{aligned}$$

A symmetrical argument applies to the second phase of the paths, which contributes an additional factor of 2, concluding the proof. ⊠

Note that without increasing the expected congestion, we can always remove any cycles in a path, so without loss of generality, we will assume that the paths are acyclic. We now obtain a concentrated result on the congestion using a straightforward Chernoff bounding argument and the fact that every packet selects its path independently of every other packet. To simplify the presentation, we give the result for constant γ, α, β, R in which case Theorem 2 gives

$$E[C(v)] = O\left(C^*_{node} \cdot (\Delta_Q^3 + (1+\Delta_Q)\log(D^* + \Delta_Q))\right).$$

The general case can be handled similarly. We have the following theorem.

Theorem 3. *When γ, α, β, R are constants, the node congestion is*

$$C_{node} = O\left(C^*_{node} \cdot (1+\Delta_Q^3) \cdot \log(n+\Delta_Q)\right),$$

with high probability.

Proof: Let $X_i = 1$ if path $p(s_i, t_i)$ uses node v, and $X_i = 0$ otherwise. Then, by Theorem 2, there is a constant A such that

$$\begin{aligned}E[C(v)] &= E[\sum_i X_i] \\ &\leq A \cdot C^*_{node} \\ &\quad \cdot \left(\Delta_Q^3 + (1+\Delta_Q)\log(D^* + \Delta_Q)\right) \\ &\leq A \cdot C^*_{node} \\ &\quad \cdot \left(\Delta_Q^3 \log n + (1+\Delta_Q)\log(n(D^* + \Delta_Q))\right) \\ &:= B.\end{aligned}$$

Let $\kappa > 2e$. Since $\sum_i X_i$ is a sum of independent Bernoulli trials, by applying a Chernoff bound [23] we obtain

$$P[C(v) > \kappa B] < 2^{-\kappa B} \leq 1/n^{\kappa A},$$

where we used the facts that $C^*_{node}, D^* \geq 1$ and $\Delta_Q \geq 0$. Taking a union bound over the n nodes multiplies by an additional n, reducing the exponent on the right to $\kappa A - 1$. Choosing a large enough κ, and noting that $D^* = O(n)$, we obtain the theorem. ⊠

13.2.2.3 Edge Congestion Analysis

For the edge congestion, the proof is similar to the node congestion. In order to carry through the same analysis, we need an upper bound on the number of edges in the area, so we can get a lower bound on the average edge congestion. If the maximum degree (maximum number of edges adjacent per node) in the network is δ, then the maximum number of edges is at most a factor of δ times the maximum number of nodes. Therefore, the result is that the optimal edge congestion is at most a factor of δ smaller than the optimal node congestion, giving the following theorem for the expected edge congestion,

Theorem 4. *Let δ be the maximum node degree. The expected congestion on an edge e is*

$$E[C(e)] \leq \delta \cdot f(\gamma, \alpha, \beta, R, \Delta_Q, D^*) \cdot C^*_{edge}.$$

A concentrated result can also be obtained for the edge congestion.

Theorem 5. *When γ, α, β, R are constants, the edge congestion is*

$$C_{edge} = O\left(\delta \cdot C^*_{edge} \cdot (1 + \Delta_Q^3) \cdot \log(n + \Delta_Q)\right),$$

with high probability.

13.2.3 Applications of Geometric Networks

The oblivious algorithm for geometric networks has applications in the 2-dimensional mesh and also in uniformly distributed unit disc graphs. The 2-dimensional mesh is an $\sqrt{n} \times \sqrt{n}$ grid of nodes, where each node is connected with at most 4 adjacent neighbors (see Figure 13.2). The nodes are placed at a unit distance from each other, and thus $R = 1/\sqrt{2}$. The rectagular area $\mathscr{A}$ is a square defined by the border nodes of the mesh so the pseudo-convexity $\gamma = 1/2$. For the default path between a pair of nodes, we choose the shortest path that connects the nodes which is closest to the geodesic and therefore $\mathsf{deviation}(Q) \leq 1/\sqrt{2}$. Since the default paths are shortest paths, $\mathsf{stretch}(Q) = 1$. Since adjacent nodes cannot be further than a unit distance, we have that $\alpha = 1$. Moreover, the number of nodes used per unit distance in the shortest path is maximized when the geodesic between the nodes is 45 degrees, which gives $\beta = \sqrt{2}$. Since the maximum node degree is 4, using Theorems 1, 3 and 5, we obtain:

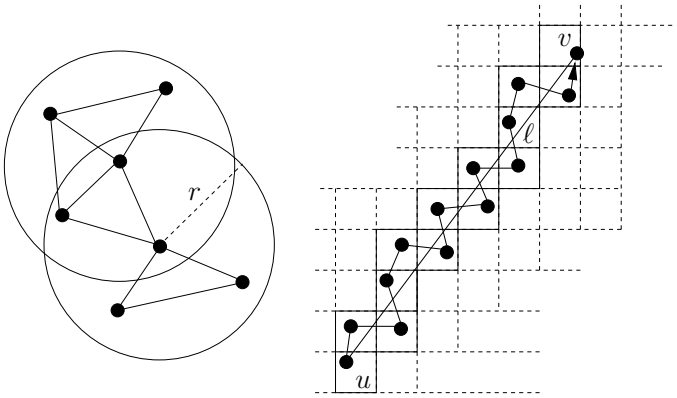

Fig. 13.6 Connectivity of a disc graph, and default path construction.

Theorem 6. *The oblivious algorithm on the mesh has* $\mathsf{stretch}(P) < 2\sqrt{2}$*, and node-congestion* $O(C^*_{node} \cdot \log n)$ *and edge-congestion* $O(C^*_{edge} \cdot \log n)$ *with high probability.*

We consider *uniform disc graphs* with n nodes distributed in an $s_1 \times s_2$ rectangle area $\mathscr{A}$, with constant pseudo-convexity $\gamma = \min\{s_1, s_2\}/2\max\{s_1, s_2\}$ (i.e., the sides are proportional to each other). In a disc graph, each node has a constant radius r and is connected to any node within this radius (see figure 13.6). We set the radius $r = 2\sqrt{2}$, and assume that no two nodes are placed within a constant distance l of each other. We consider a uniform distribution for the nodes in the area, i.e., the area is divided into non-overlapping unit squares, and every unit square area contains a number of nodes between 1 and $k = O(\frac{1}{l^2})$ nodes, where k is a constant. By the choice of r, two nodes within the same square or in adjacent squares will be connected. Thus, $R \leq \sqrt{2}$, and since there are at most 32 squares containing nodes which could possibly be adjacent to a particular node, the maximum node degree is bounded by $\delta \leq 32k$.

We now explain how to construct the default paths (see Figure 13.6). Consider two nodes u and v in area $\mathscr{A}$ and construct the line ℓ that connects $\mathbf{x}(u)$ to $\mathbf{x}(v)$. This line passes through a collection of unit squares, forming a path with adjacent unit squares. We pick one node from each square and construct the default path by connecting these nodes. Since for every node in the path, the line passes through the corresponding unit square containing the node, $\mathsf{deviation}(Q) \leq \sqrt{2}$. The number of unit squares in the formation of the default path is no more than $2|\ell|$, so the longest default path consists of at most $2|\ell|$ nodes. The shortest path has to use at least $|\ell|/r$ nodes; therefore, $\mathsf{stretch}(Q) \leq 2r$. Since $\mathsf{dist}_G(u,v) \geq \mathsf{dist}_E(u,v)/r$, $\alpha \geq 1/r$. If $\mathsf{dist}_G(u,v) = 1$, $\mathsf{dist}_E(u,v) \geq l$, so $\mathsf{dist}_G(u,v)/\mathsf{dist}_E(u,v) \leq \frac{1}{l}$. More generally, we know that $\mathsf{dist}_G(u,v) \leq 2|\ell|$ since the default path has $2|\ell|$ hops, so the shortest path cannot have more. Thus, $\mathsf{dist}_G(u,v)/\mathsf{dist}_E(u,v) \leq \max\{2, 1/l\}$, so $\beta \leq \max\{2, 1/l\}$, Applying Theorems 1, 3 and 5, we obtain:

Theorem 7. *On uniform disc graphs, the oblivious routing algorithm has* $\mathsf{stretch}(P) = O(1)$*, and node-congestion* $O(C^*_{node} \cdot \log n)$ *and edge-congestion* $O(C^*_{edge} \cdot \log n)$ *with a high probability.*

13.3 Mesh Networks

13.3.1 Preliminaries on Mesh Networks

The d-dimensional mesh M is a d-dimensional grid of nodes with side length m_i in dimension i. There is a link connecting a node with each of its $2d$ neighbors (except for the nodes at the boundaries of the mesh). We denote by n the size of M, $n = \mathsf{size}(M) = \prod_{i=1}^{d} m_i$, and by $|E|$ the number of edges in the network. Each node has a coordinate. For example, in the 2 dimensional mesh, the top-left node has coordinate $(0,0)$. We refer to specific submeshes by giving its end points in every dimension, for example, $[0,3][2,5]$ refers to a 4×4 submesh, with the x coordinate ranging from 0 to 3 and the y coordinate from 2 to 5.

The input for the path selection problem is a set of N sources and destinations (i.e. packets), $\Pi = \{s_i, t_i\}_{i=1}^{N}$ and the mesh M. The output is a set of paths, $P = \{p_i\}$, where each path $p_i \in P$ is from node s_i to node t_i. The length of path p, denoted $|p|$, is the number of edges it uses. We denote the length of the shortest path from s to t by $\mathsf{dist}(s,t)$. We will denote by D^* the maximum shortest distance, $\max_i \mathsf{dist}(s_i,t_i)$. The *stretch* of a path p_i, denoted $\mathsf{stretch}(p_i)$, is the ratio of the path length to the shortest path length between its source and destination, $\mathsf{stretch}(p_i) = |p_i|/\mathsf{dist}(s_i,t_i)$. The *stretch factor* for the collection of paths P, denoted $\mathsf{stretch}(P)$, is the maximum stretch of any path in P, $\mathsf{stretch}(P) = \max_i \mathsf{stretch}(p_i)$.

For a submesh $M' \subseteq M$, let $\mathsf{out}(M')$ denote the number of edges at the boundary of M', which connect nodes in M' with nodes outside M'. For any routing problem Π, we define the *boundary congestion* as follows. Consider some submesh of the network M'. Let Π' denote the packets (pairs of sources and destinations) in Π which have either their source or destination in M', but not both. All the packets in Π' will cross the boundary of M'. The paths of these packets will cause congestion at least $|\Pi'|/\mathsf{out}(M')$ times. We define the boundary congestion of M' to be $B(M',\Pi) = |\Pi'|/\mathsf{out}(M')$. For the routing problem Π, the boundary congestion B is the maximum boundary congestion over all its submeshes, i.e. $B = \max_{M' \subseteq M} B(M',\Pi)$. Clearly, $C^* \geq B$.

13.3.2 Oblivious Routing on 2-Dimensional Mesh Networks

Here we show how to select the paths in a 2-dimensional mesh with equal side lengths $m = 2^k,\ k \geq 0$. We consider this case here for expository ease, however

the result generalizes to the case of unequal side lengths which are not necessarily powers of 2. We use the 2-dimensional case to illustrate the main ideas, before generalizing to the d-dimensional case in the next section. The path selection algorithm relies on a decomposition of the mesh to submeshes, and then constructing an access graph, as we describe next.

13.3.2.1 Decomposition to Submeshes

We decompose the mesh M into two types of submeshes, type-1 and type-2, as follows.

- *Type*-1 *Submeshes:* We define the type-1 submeshes recursively. There are $k+1$ levels of type-1 submeshes, $\ell = 0,\ldots,k$. The mesh M itself is the only level 0 submesh. Every submesh at level ℓ can be partitioned into 4 submeshes by dividing each side by 2. Each resulting submesh is a type-1 submesh at level $\ell+1$. This construction is illustrated in Figure 13.7. In general, at level ℓ there are $2^{2\ell}$ submeshes each with side $m_\ell = 2^{k-\ell}$. Note that the level k submeshes are the individual nodes of the mesh.
- *Type*-2 *Submeshes:* There are $k-1$ levels of type-2 submeshes, $\ell = 1,\ldots,k-1$. The type-2 submeshes at level ℓ are obtained by first extending the grid of type-1 meshes by adding one layer of type-1 meshes along every dimension. The resulting grid is then translated by the vector $-(m_\ell/2, m_\ell/2)$. In this enlarged and translated grid, some of the resulting translated submeshes are entirely within M. These are the *internal* type-2 submeshes. For the remaining *external* type-2 submeshes, we keep only their intersection with M, except that we discard all the "corner" submeshes, because they will be included in the type-1 submeshes at the next level. Notice that all the type-2 submeshes have at least 1 side with a length of m_ℓ nodes. Figure 13.7 illustrates the construction.

A submesh of M is *regular* if it is either type-1 or type-2. Unless otherwise stated, a submesh will always refer to regular submeshes. The following lemma follows from the construction of the regular submeshes.

Lemma 6. *The mesh decomposition satisfies the following properties.*

*(1) The type-*1 *submeshes at a given level are disjoint, as are the type-*2 *submeshes.*
*(2) Every regular submesh at level ℓ can be partitioned into type-*1 *submeshes at level $\ell+1$.*
(3) Every regular submesh at level $\ell+1$ is completely contained in a submesh at level ℓ of either type-1 or type-2, or both.

13.3.2.2 Access Graph

The access graph $G(M)$, for the mesh M, is a leveled graph with $k+1$ levels of nodes, $\ell = 0,\ldots,k$. The nodes in the access graph correspond to the distinct regular

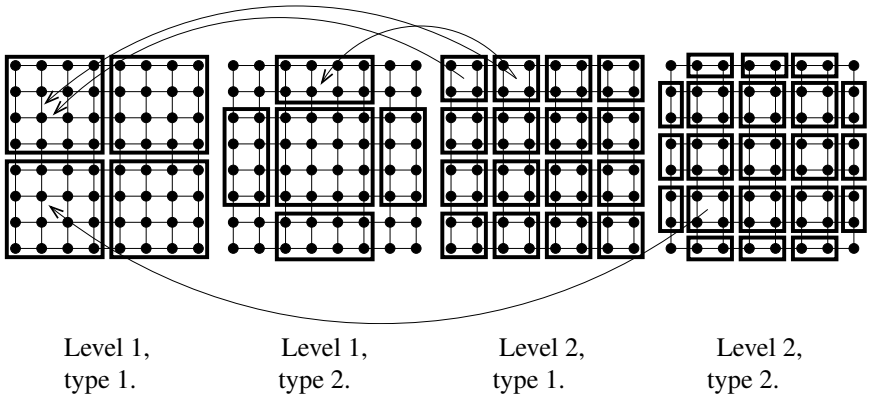

Fig. 13.7 Mesh decomposition for the $2^3 \times 2^3$ mesh. Arrows indicate the parents of a submesh.

submeshes. Specifically, every level-ℓ submesh (type-1 or type-2) corresponds to a level ℓ node in $G(M)$. Edges exist only between adjacent levels of the graph. Let u_ℓ, $u_{\ell+1}$ be level ℓ and $\ell+1$ nodes of $G(M)$ respectively. The edge $(u_\ell, u_{\ell+1})$ exists if the regular submesh corresponding to u_ℓ completely contains the regular submesh corresponding to $u_{\ell+1}$. We borrow some terminology from trees. We say that u_ℓ is a *parent* of $u_{\ell+1}$ in $G(M)$; the parent relationship is illustrated in Figure 13.7, for the corresponding submeshes. Note that the access graph is not necessarily a tree, since a node can have two parents (a consequence of Lemma 6, part (3)). The depth of a node is the same as its level ℓ, and its height is $k-\ell$. Nodes at height 0 have no children, and are referred to as leaves. The leaves in $G(M)$ correspond to single nodes in the mesh. There is a unique root at level 0, which corresponds to the whole mesh M.

Let $p = (u_1, u_2, \ldots, u_k)$ be a path in $G(M)$. We say that p is *monotonic* if every node is of increasing level (i.e., the level of u_i is higher than the level of u_{i+1}), and the respective submeshes of nodes $u_2, \ldots, u_k$ are all of type-1. If p is monotonic, then we say that u_1 is an *ancestor* of u_k. We will use a function g to map nodes in the access graph to submeshes. Let u be a node in the access graph with a corresponding submesh M'. We define the function g so that $g(u) = M'$. Denote by g^{-1} the inverse of function g, that is, $g^{-1}(M') = u$. Using induction on the height of $G(M)$, and part (2) of Lemma 6, we obtain the following lemma:

Lemma 7. *Let v be any node (1×1 submesh) of a regular submesh $M' \subseteq M$, then $g^{-1}(M')$ is an ancestor of $g^{-1}(v)$.*

Let u and v be two leaves of $G(M)$, and let A be their (not necessarily unique) deepest common ancestor; note that A exists and in the worst case is $g^{-1}(M)$ (a consequence of Lemma 7). Let $p = (u, \ldots, A, \ldots, v)$, be the concatenation of two monotonic paths, one from A to u and the other from A to v. We will refer to p as the *bitonic* path between u and v. Submesh $g(A)$ may be type-1 or type-2, all the

other submeshes in p are of type-1. We will refer to $g(A)$ as a "bridge" submesh, since it provides the connecting point between two monotonic paths. Note that type-2 submeshes can be used as bridges between type-1 submeshes, when constructing bitonic paths between leaves. Further, only one type-2 submesh is ever needed in a bitonic path. These access graph paths will be used by the path selection algorithm. Suppose that $height(A) = h_A$. The length of a bitonic path from u to v is $2h_A$. We now show that h_A cannot be too large. This will be important in proving that the path selection algorithm gives constant stretch.

Lemma 8. *The deepest common ancestor of two leaves u and v has a height of at most* $\lceil \log \mathsf{dist}(g(u), g(v)) \rceil + 2$.

Proof: Let $s,t \in M$ such that $s = g(u)$ and $t = g(v)$. We show that there is a common ancestor with height at most $\lceil \log \mathsf{dist}(s,t) \rceil + 2$.

Assume, first, that instead of a mesh, the network is a torus (the same result holds for the mesh, with a minor technical detail in the proof due to edge effects, which we will discuss later). In this case, all type-2 meshes are of the same size. We obtain the regular submeshes in the original mesh after truncation of the submeshes at the borders of the torus. Note that all distances, however, are measured on the mesh.

Let $\mu = 2^{\lceil \log \mathsf{dist}(s,t) \rceil} \geq \mathsf{dist}(s,t)$. If $4\mu \geq 2^k$, then the root, $g^{-1}(M)$, is a common ancestor with, at most, a height of $\lceil \log \mathsf{dist}(s,t) \rceil + 2$, so assume that $4\mu < 2^k$. Node s is contained in some type-1 submesh of side length 4μ. Without loss of generality (since we are on a torus), assume that this submesh is $M_1 = [0, 4\mu - 1]^2$. If M_1 also contains t, then we are done, since by Lemma 7, $g^{-1}(M_1)$ is a common ancestor at height $\lceil \log \mathsf{dist}(s,t) \rceil + 2$. So suppose that t is contained in some other (adjacent) type-1 submesh M_2. There are two possibilities for M_2.

1. M_1 and M_2 are diagonaly adjacent, so without loss of generality, let $M_2 = [4\mu, 8\mu - 1][4\mu, 8\mu - 1]$. Since $\mathsf{dist}(s,t) \leq \mu$, $s \in [3\mu, 4\mu - 1]^2$ and $t \in [4\mu, 5\mu - 1]^2$, and so the type-2 submesh $[2\mu, 6\mu - 1]^2$ contains both s and t.
2. M_2 is laterally adjacent to M_1, so, without loss of generality, let M_2 be to the right of M_1, i.e. $M_2 = [0, 4\mu - 1][4\mu, 8\mu - 1]$. In this case, s must be in the right half of M_1 and t in the left half of M_2, i.e. $s \in [0, 4\mu - 1][3\mu, 4\mu - 1]$ and $t \in [0, 4\mu - 1][4\mu, 5\mu - 1]$. There are four cases:
   a. $s \in [0, 2\mu - 1][3\mu, 4\mu - 1]$ and $t \in [0, 2\mu - 1][4\mu, 5\mu - 1]$, in which case the type-2 submesh $[-2\mu, 2\mu - 1][2\mu, 6\mu - 1]$ contains s,t;
   b. $s \in [2\mu, 4\mu - 1][3\mu, 4\mu - 1]$ and $t \in [2\mu, 4\mu - 1][4\mu, 5\mu - 1]$, in which case the type-2 submesh $[2\mu, 6\mu - 1]^2$ contains s,t;
   c. $s \in [\mu, 2\mu - 1][3\mu, 4\mu - 1]$ and $t \in [2\mu, 3\mu - 1][4\mu, 5\mu - 1]$, in which case the type-2 submesh $[\mu, 3\mu - 1][3\mu, 5\mu - 1]$ at height $\lceil \log \mathsf{dist}(s,t) \rceil + 1$ contains s,t;
   d. $s \in [2\mu, 3\mu - 1][3\mu, 4\mu - 1]$ and $t \in [\mu, 2\mu - 1][4\mu, 5\mu - 1]$, which is similar to (c).

In all cases, s and t are contained in a submesh of a height of at most $\lceil \log \mathsf{dist}(s,t) \rceil + 2$.

Algorithm 12 Oblivious Routing for 2-Dimensional Mesh Networks

Input: Source s and destination t in the mesh M;
Output: Path $p(s,t)$ from s to t in M;

```
Let (u_0, ..., u_l) denote a bitonic path in G(M) from g^{-1}(s) to g^{-1}(t);
for i = 0 to l do
    Select a node v_i in g(u_i) uniformly at random; // v_0 = s and v_l = t
    if 1 ≤ i ≤ l then
        Construct subpath r_i from v_{i-1} to v_i by picking a dimension by dimension shortest path [1] (an at most one-bend path), according to a random ordering of the dimensions;
    end if
end for
The path p(s,t) is obtained by concatenating the subpaths r_i, p(s,t) = r_0 r_1 ··· r_{l-1};
```

To complete the argument, we now suppose that the network is a mesh (instead of a torus). If the ancestor submesh constructed in the torus is also a regular submesh in the mesh, then there is nothing to prove. So, assume that the ancestor constructed in the torus is not a regular submesh of the mesh. In particular, the ancestor constructed on the torus must be composed of two type-2 submeshes, on opposite sides of the mesh. If s,t are both contained in one of these submeshes, then they are both contained in a type-2 submesh of a height of at most $\lceil \log \mathsf{dist}(s,t) \rceil + 2$. The only remaining case is that s is in one of these submeshes and t is in the other. In this case, $\mathsf{dist}(s,t) \geq 2^{k-1}$, and since $\mu \geq \mathsf{dist}(s,t)$, we have that $\mu \geq 2^{k-1}$, or that $4\mu \geq 2^{k+1}$ which contradicts the assumption that $4\mu < 2^k$, concluding the proof. ⊠

13.3.2.3 Path Selection

Given the access graph, the procedure to determine a path from a given source s to a destination t is summarized in Algorithm 12. Note that the algorithm is oblivious and local, since each source-destination pair can obtain a path independently of the other paths. We will now show that our algorithm with the generalized access graph, in addition to obtaining optimal congestion, also controls the stretch. First we show the constant stretch property of the selected paths.

Theorem 8. *For any two distinct nodes s,t, $\mathsf{stretch}(p(s,t)) \leq 64$.*

Proof: Let h be the height of the deepest common ancestor of s and t. Then $p(s,t)$ is the concatenation of paths constructed from the dimension by dimension paths in meshes of sides $2^1,\ldots,2^{h-1},2^h,2^{h-1},\ldots,2^1$. A path in a mesh of side ℓ has length of at most $2\ell-1$, so by adding the lengths of these paths, we have that $|p(s,t)| \leq 2(2^1+\cdots+2^h+2^h+\cdots+2^1-2h)$ which implies that $|p(s,t)| \leq 2^{h+3}-4h$. Since

s and t are distinct, $h \geq 1$. By Lemma 8, $h \leq \log \mathsf{dist}(s,t) + 3$, and the theorem follows. ⊠

We now relate the congestion of the paths selected to the optimal congestion C^*. Let e denote an edge in M. Let $C(e)$ denote the load on e, i.e., the number of times that edge e is used by the paths of all the packets. We will get an upper bound on $E[C(e)]$, and then, using a Chernoff bound, we will obtain a concentrated result.

We start by bounding the probability that some particular subpath formed by the path selection algorithm uses edge e. Consider the formation of a subpath r_i from a submesh M_1 to a submesh M_2, such that M_2 completely contains M_1, and e is a member of M_2. According to the path selection algorithm, mesh M_1 is of type-1, thus all of its sides are equal to m_ℓ, where ℓ is the level of M_1. We show the following lemma.

Lemma 9. *Subpath r_i uses edge e with probability at most $2/m_\ell$.*

Proof: For subpath r_i, let v_1 be the starting node in M_1 and v_2 the ending node in M_2. Suppose $e = (v_3, v_4)$. Without loss of generality, suppose e is vertical. Since the subpath is a one-bend path, edge e can be used only when either v_1 or v_2 have the same x coordinate as e. This event occurs at a probability of at most $2/m_\ell$. ⊠

Let P' be the set of paths that go from M_1 to M_2 or vice-versa. Let $C'(e)$ denote the congestion that the packets P' cause on e. We show:

Lemma 10. $E[C'(e)] \leq 2|P'|/m_\ell$.

Proof: We can write $P' = P_1 \cup P_2$, where P_1 is the set of subpaths from M_1 to M_2, and P_2 is the subpaths from M_2 to M_1. Then, from Lemma 9, the expected congestion on edge e due to the subpaths in P_1 is bounded by $2|P_1|/m_\ell$. Using a similar analysis, the expected congestion on e due to subpaths in P_2 is bounded by $2|P_2|/m_\ell$. Since the congestion on e due to the paths in P' is the sum of the congestions due to P_1 and P_2, we obtain $E[C'(e)] \leq 2(|P_1| + |P_2|)/m_\ell = 2|P'|/m_\ell$. ⊠

From the definition of the boundary congestion, we have that $B \geq B(M_1, \Pi) \geq |P'|/\mathsf{out}(M_1)$. Therefore, $C^* \geq |P'|/\mathsf{out}(M_1)$. Since each side of M_1 has m_ℓ nodes, we have that $\mathsf{out}(M_1) \leq 4m_\ell$. From Lemma 10, we therefore obtain:

Lemma 11. $E[C'(e)] \leq 8C^*$.

We "charge" this congestion to submesh M_2. By Lemma 8, only submeshes up to height $h < \log D^* + 3$ can contribute to the congestion on edge e (submeshes of type-1). By summing the congestions due to these at most $2(\log D^* + 3)$ submeshes (a type-1 and a type-2 submesh at each level), and by using Lemma 11, we arrive at an upper bound for the expected congestion on edge e:

Lemma 12. $E[C(e)] \leq 16C^*(\log D^* + 3)$.

Note that without increasing the expected congestion, we can always remove any cycles in a path, so without loss of generality, we will assume that the paths obtained are acyclic. We now obtain a concentrated result on the congestion C obtained by our algorithm, using the fact that every packet selects its path independently of every other packet.

Theorem 9. *$C = O(C^* \log n)$ with high probability.*

Proof: Let $X_i = 1$ if path p_i uses edge e, and 0 otherwise. Then $E[C(e)] = E[\sum_i X_i] \leq 16C^*(\log D^* + 3)$. Let $|E|$ be the number of edges in the mesh. For $|E| > 8$, $E[C(e)] \leq 16C^* \log(|E|D^*)$. Let $\kappa > 2e$, then applying a Chernoff bound [23], and using the fact that $C^* \geq 1$ we find that $P[C(e) > 16\kappa C^* \log(|E|D^*)] < (|E|D^*)^{-16\kappa}$. Taking a union bound over all the edges, we obtain

$$P[\max_{e \in E} C(e) > 16\kappa C^* \log(|E|D^*)] < \frac{1}{(|E|D^*)^{16\kappa - 1}}.$$

Using the fact that $D^* = O(|E|)$, $|E| = O(n^2)$, and choosing $\kappa = 2e + 1$, we get $C = O(C^* \log n)$ with high probability. ⊠

Acknowledgements We are grateful to the reviewers of this book chapter.

References

1. J. Aspens, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts. Online load balancing with applications to machine scheduling and virtual circuit routing. In *Proceedings of the 25th ACM Symposium on Theory of Computing*, pages 623–631, 1993.
2. F. Meyer auf der Heide, C. Schindelhauer, K. Volbert, and M. Grünewald. Congestion, dilation, and energy in radio networks. *Theory of Computing Systems*, 37(3):343–370, 2004.
3. B. Awerbuch and Y. Azar. Local optimization of global objectives: competitive distributed deadlock resolution and resource allocation. In *Proceedings of 35th Annual Symposium on Foundations of Computer Science*, pages 240–249, Santa Fe, New Mexico, 1994.
4. Y. Azar, E. Cohen, A. Fiat, H. Kaplan, and H. Racke. Optimal oblivious routing in polynomial time. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 383–388, San Diego, CA, June 2003. ACM Press.
5. Marcin Bienkowski, Miroslaw Korzeniowski, and Harald Räcke. A practical algrorithm for constructing oblivious routing schemes. In *Proceedings of the 15th Annual ACM Symposium on Parallelism in Algorithms and Architectures*, pages 24–33, Jun. 2003.
6. A. Borodin and J. E. Hopcroft. Routing, merging, and sorting on parallel models of computation. *Journal of Computer and System Science*, 30:130–145, 1985.
7. Costas Busch, Malik Magdon-Ismail, and Jing Xi. Oblivious routing on geometric networks. In *Proceedings of the 17th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 316–324, Las Vegas, Nevada, July 2005.
8. Costas Busch, Malik Magdon-Ismail, and Jing Xi. Optimal oblivious path selection on the mesh. *IEEE Transactions on Computers*, 57(5):660–671, May 2008.
9. Ioannis Chatzigiannakis, Tassos Dimitriou, Sotiris Nikoletseas, and Paul Spirakis. A probabilistic algorithm for efficient and robust data propagation in wireless sensor networks. *Ad Hoc Networks*, 4(5):621 – 635, 2006.

10. Ioannis Chatzigiannakis, Sotiris Nikoletseas, and Paul G. Spirakis. Efficient and robust protocols for local detection and propagation in smart dust networks. *Mob. Netw. Appl.*, 10(1-2):133–149, 2005.
11. Shlomi Dolev, Ted Herman, and Limor Lahiani. Polygonal broadcast, secret maturity, and the firing sensors. *Ad Hoc Networks*, 4(4):447 – 486, 2006.
12. Shlomi Dolev and Nir Tzachar. Empire of colonies: Self-stabilizing and self-organizing distributed algorithm. *Theor. Comput. Sci.*, 410(6-7):514–532, 2009.
13. Charilaos Efthymiou, Sotiris Nikoletseas, and Jose Rolim. Energy balanced data propagation in wireless sensor networks. *Wirel. Netw.*, 12(6):691–707, 2006.
14. Jie Gao and Li Zhang. Tradeoffs between stretch factor and load balancing ratio in routing on growth restricted graphs. In *PODC '04: Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pages 189–196, New York, NY, USA, 2004.
15. Chris Harrelson, Kristen Hildrum, and Satish Rao. A polynomial-time tree decomposition to minize congestion. In *Proceedings of the 15th Annual ACM Symposium on Parallelism in Algorithms and Architectures*, pages 34–43, Jun. 2003.
16. Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, John Heidemann, and Fabio Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. Netw.*, 11(1):2–16, 2003.
17. Christos Kaklamanis, Danny Krizanc, and Thanasis Tsantilas. Tight bounds for oblivious routing in the hypercube. In *Proceedings of 2nd IEEE Symposium on Parallel and Distributed Processing (2nd SPAA 90)*, pages 31–36, Crete, Greece, July 1990.
18. F. T. Leighton, B. M. Maggs, and S. B. Rao. Packet routing and job-scheduling in $O(congestion + dilation)$ steps. *Combinatorica*, 14:167–186, 1994.
19. F. Thomson Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays - Trees - Hypercubes*. Morgan Kaufmann, San Mateo, 1992.
20. Tom Leighton, Bruce Maggs, and Andrea W. Richa. Fast algorithms for finding O(congestion + dilation) packet routing schedules. *Combinatorica*, 19:375–401, 1999.
21. B. M. Maggs, F. Meyer auf der Heide, B. Vöcking, and M. Westerman. Exploiting locality in data management in systems of limited bandwidth. In *Proceedings of the 38th Annual Symposium on the Foundations of Computer Science*, pages 284–293, 1997.
22. Friedhelm Meyer auf der Heide and Berthold Vöcking. Shortest-path routing in arbitrary networks. *Journal of Algorithms*, 31(1):105–131, April 1999.
23. Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, Cambridge, UK, 2000.
24. Rafail Ostrovsky and Yuval Rabani. Universal $O(\text{congestion+dilation+}\log^{1+\varepsilon} N)$ local control packet switching algorithms. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, pages 644–653, New York, May 1997.
25. Lucian Popa, Afshin Rostamizadeh, Richard Karp, Christos Papadimitriou, and Ion Stoica. Balancing traffic load in wireless networks with curveball routing. In *MobiHoc*, 2007.
26. Harald Räcke. Minimizing congestion in general networks. In *Proceedings of the 43rd Annual Symposium on the Foundations of Computer Science*, pages 43–52, Nov. 2002.
27. Harald Räcke. *Data management and routing in general networks*. Phd thesis, University of Paderborn, 2003.
28. Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proceedings of the 40th STOC*, pages 255–264, 2008. Co-Winner of Best Paper Award.
29. P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365–374, 1987.
30. C. Scheideler. Course notes. `http://www14.in.tum.de/lehre/2005WS/na/index.html.en`.
31. A. Srinivasan and C-P. Teo. A constant factor approximation algorithm for packet routing, and balancing local vs. global criteria. In *Proceedings of the ACM Symposium on the Theory of Computing (STOC)*, pages 636–643, 1997.
32. L. G. Valiant. A scheme for fast parallel communication. *SIAM Journal on Computing*, 11:350–361, 1982.

33. L. G. Valiant and G. J. Brebner. Universal schemes for parallel communication. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing*, pages 263–277, May 1981.
34. Feng Zhao and Leonidas J. Guibas. *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann, 2004.