Optimal Oblivious Routing in Hole-Free Networks

Costas Busch[1] and Malik Magdon-Ismail[2]

[1] Louisiana State University, Baton Rouge, LA 70803, USA; busch@csc.lsu.edu
[2] Rensselaer Polytechnic Institute, Troy, NY 12180, USA; magdon@cs.rpi.edu

Abstract. We study oblivious routing algorithms in which the packet paths are constructed independently of each other. Oblivious algorithms are inherently distributed and they can be designed to efficiently balance the network utilization. We give an oblivious routing algorithm for the class of hole-free networks, in which the nodes are topologically embedded in simple areas of the plane. Such networks appear frequently in wireless and sensor network topologies. The algorithm achieves optimal congestion and stretch. The stretch of the resulting paths is constant. The congestion is $O(C^* \log n)$, where C^* is the optimal non-oblivious congestion and n is the number of nodes. This congestion bound is asymptotically worst-case optimal for oblivious routing algorithms.

Key words: oblivious routing, congestion, path stretch, wireless networks, sensor networks

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 selected 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. The objective of this work is to present oblivious algorithms with low congestion and small path stretch. For congestion we consider the bottleneck metric C which is equal to the maximum number of selected paths that use any edge in the network.

Oblivious routing is applicable to wireless and sensor networks. It is particularly suitable to energy and power constraint networks (e.g. battery operated nodes), since it can help to extend the time until some node runs out of power. Lowering the network congestion (lowering edge bottlenecks), results to improved load balancing and thus prolonged lifetime and better utilization of the network. In addition, paths of small stretch (ratio of path length to shortest path) result to low overall energy utilization. Oblivious algorithms are also easy to implement in wireless and sensor networks, on account of their simplicity.

We give an oblivious routing algorithm designed for *hole-free* networks which are suitable to model wireless network communication environments. A hole-free

QSHINE 2010, November 17-19, Houston, United States

DOI 10.4108/eai.17-11-2010.2245395

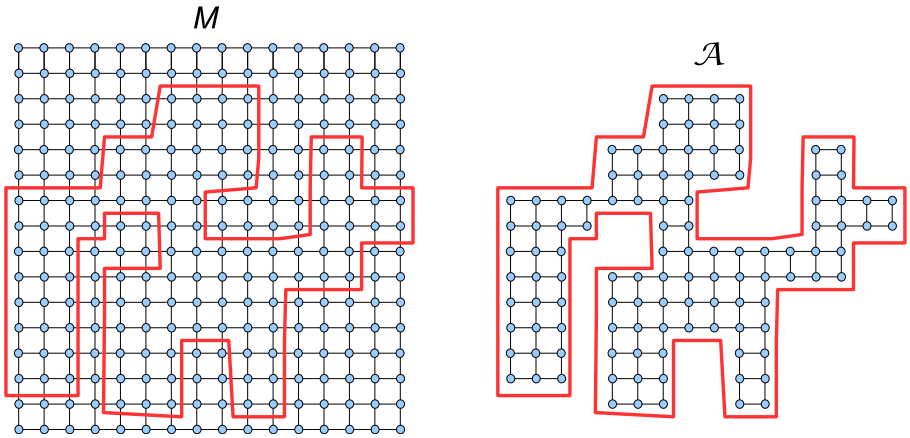


Fig. 1. Hole-free grid-like network G

network is embedded in the 2-dimensional Euclidian plane inside a *simple area* $\mathcal{A}$ (see Figure 1). We are considering *grid-like* networks, which are induced when we apply a simple area $\mathcal{A}$ on top of a $m \times m$ grid M. Hole-free grid-like graphs are interesting because they can model wireless and sensor network topologies where each node is connected with a link with its neighbors at distance 1, within the boundaries of area $\mathcal{A}$.

We give an oblivious routing algorithm on any hole-free grid-like graph G that gives paths with optimal stretch and congestion. In particular, given a set of packets Π with respective sources and destinations, our algorithm returns a set of paths P, one path for each packet in Π, such that $\mathsf{stretch}(P) = \Theta(1)$. In other words, the length of every resulting path is within a constant factor from the length of the respective shortest path. Further, the congestion of paths P is $O(C^* \log n)$, where n is the number of nodes in G, and C^* denotes the optimal congestion that can be achieved for the packets Π. The upper bound is optimal since it is known that any oblivious algorithm has congestion $\Omega(C^* \log n)$ in the worst case for grids [13].

The algorithm we give is oblivious and randomized, which means that each path is computed with some randomized choices in a specific manner that does not depend on other path requests in the network. In particular, a path from a source s to a destination t is computed by using a shortest path q based on which we estimate a sequence of adjacent squares along q. The requested path from s to t is formed by concatenating randomized one-bend paths or two-bend paths formed in each square in the sequence. The squares in the sequence are selected from a hierarchical partition of the grid M into canonical squares of various sizes placed in fixed positions in M. A crucial aspect of the algorithm is that the sum of the side lengths of the squares in the sequence is within a constant factor to the length of the shortest path from s to t. This helps to control the stretch. The

congestion is controlled by adjusting the square sizes to the cut sizes (between s and t) which involve the nodes in the squares.

1.1 Related Work

Valiant and Brebner [21, 20] are the first to propose the oblivious routing technique of using a single random intermediate node in order to minimize congestion in networks. They give an appropriate general approach for oblivious routing based on solutions to flow problems in the network. Applications are permutation routing problems on the hypercube and butterfly networks.

The motivation for minimizing congestion and stretch simultaneously is because there exist packet scheduling algorithms [12] which deliver the packets along the given paths in time very close to the optimal $O(C + D)$, where D is the maximum path length of the routing algorithm. A trivial lower bound for the total time to transfer all the packets along the selected paths is $\Omega(C+D)$. Hence $C + D$ is a natural metric by which to measure the quality of the paths that are produced by a routing algorithm. Our oblivious routing algorithm provides paths with small $C + D$ because of the low congestion and small stretch.

Maggs *et al.* [13] give an oblivious algorithm for the d-dimensional *mesh* (grid) with congestion $O(d \cdot C^* \cdot \log n)$. However, the stretch factor in that algorithm is unbounded. To control stretch within a constant factor, we generalize in [8] the hierarchical decomposition for the mesh denoted by an *access tree* [13] to a more general *access graph*. Oblivious congestion minimizing algorithms which control the stretch have been also considered by Scheideler [18] for the 2-dimensional mesh, which uses a different approach by routing within a square containing the source and destination and building an access tree specific to this square.

Following the work in [13], there have been extensions to general networks [4, 5, 10, 15, 16], where progressively better oblivious algorithms with near optimal congestion. However, in all these algorithms the stretch is unbounded. Further, most of these algorithms are based on a hierarchical decomposition of the network into clusters, which requires a logarithmic number of intermediate nodes. In [7] we present an oblivious routing algorithm for geometric networks, which are special types of networks embedded in the 2-dimensional grid, and it is a very restricted subclass of hole-free networks. That algorithm uses *a single* random intermediate node and doesn't depend on any hierarchical clustering.

Lower bounds on the competitive ratio of oblivious routing has been studied for various types of networks. Maggs et al. [13] give the $\Omega(\frac{C^*}{d} \cdot \log n)$ lower bound on the competitive ratio of an oblivious algorithm on the mesh. Valiant and Brebner [21] perform a worst case theoretical analysis on oblivious routing on specific network topologies such as the hypercube. Borodin and Hopcroft [6] and Kaklamanis et al. [11] showed that deterministic oblivious routing algorithms can not approximate the minimal load on most non-trivial networks, which justifies the necessity for randomization.

Non-oblivious approaches to optimizing $C + D$ have received considerable attention, and near optimal algorithms are discussed in [1, 3, 17, 19]. As already mentioned, such offline algorithms require knowledge of the traffic distribution

a priori and generally do not scale well with the number of packets. Tradeoffs between stretch and congestion have been studied in wireless networks [9, 2].

As an alternative routing scheme for sensor networks, *curve routing* relies on geographic information, which can be obtained through GPS devices [22]. Packets are sent along specified trajectories from sources to destinations. The trajectories are defined in space, and then they are projected to actual paths in the network. Packets contain information about the path trajectory, and each time they are forwarded to the next best node that is closer to the trajectory and the destination. This has the benefit that the actual network path does not need to be precisely defined, but it can be determined on the fly in a similar way as in geographic routing. Thus, such routing methods are suitable for sensor network and wireless ad hoc networks, where the actual graph connectivity may not be known precisely but there is some information about the geographic area that contains the network. Recently, *curveball routing* has been proposed as a routing method to send packets along curves with the benefit of load-balancing the node utilization [14]. All the nodes are projected (uniformly) in the surface of a sphere. Then a routing curve is obtained from the shortest path in the sphere that connects the source to the destination. For the virtual path we only need to know the virtual coordinates of the source and destination. Note that the shortest path in the sphere may not be realized in the actual network, since we may not be able to find nodes closer to the destination in virtual coordinates. In such situations regular geographic routing may be used. Since the surface of a sphere is symmetric, the expected load on the nodes is balanced.

Paper Outline. We begin with some preliminary definitions in Section 2. We continue with describing how to construct sequences of squares for paths in Section 3. We give the oblivious routing algorithm in Section 4. We finish with its congestion and stretch analysis in Section 5.

2 Preliminaries

An area $\mathcal{A}$ on the plane is *simple* (or *hole-free*) if every closed curve in $\mathcal{A}$ can be continuously deformed into a point. Intuitively, the area bounded by any closed curve is completely contained in the area $\mathcal{A}$. For example a disc is hole free, but an annulus is not.

We now consider a 2-dimensional $m \times m$ grid (mesh) graph M consisting of nodes at positions (i, j) where $i, j \in [0, 1, \ldots, m-1]$, and $(0, 0)$ is a the bottom left, such that each node is connected with an edge to any node at distance 1 (there are at most four such nodes). On top of this grid, we draw any simple closed area $\mathcal{A}$. The *induced subgraph* of $\mathcal{A}$ is the subgraph $G_{\mathcal{A}}$ consisting of all nodes and edges which lie in the area $\mathcal{A}$ with the removal of all edges which cross the boundary of $\mathcal{A}$. If the induced subgraph $G_{\mathcal{A}}$ is connected, the we say that $G_{\mathcal{A}}$ is a *simple-area grid-like graph* (or *hole-free grid-like graph*). This construction extends the notion of hole-free from areas on the plane to graphs with natural

embeddings on the plane. From now on, G will refer to a simple-area grid-like network.

2.1 Canonical Squares

Let M be an $m \times m$ grid which contains G as a subgraph, where m is a power of 2. We can divide M into $1 + \lg m$ levels of *canonical square subgraphs* as follows. For $0 \leq \ell \leq \lg m$, the canonical square subgraphs at level ℓ partition M into $2^{2(\lg m - \ell)}$ square subgraphs. Each canonical square subgraphs at level ℓ is a $2^\ell \times 2^\ell$ grid subgraph of M, whose bottom left corner node has coordinates $(i \cdot 2^\ell, j \cdot 2^\ell)$, where $0 \leq i, j \leq 2^{\lg m - \ell} - 1$.

For simplicity, we will refer to the canonical square subgraph as canonical squares. The α-partition of M consists of the $\alpha \times \alpha$ canonical squares of M. Note that two canonical squares are either node disjoint or one contains the other. Further, every node of M is contained in exactly $1 + \lg m$ canonical squares. Two canonical squares M_1 and M_2 (not necessarily at the same level) are *adjacent* if they are disjoint and there is an edge from M_1 to M_2, i.e. an edge $(u, v) \in M$, such that $u \in M_1$ and $v \in M_2$.

Given a simple-area grid-like graph G defined in M, a canonical square B is *internal* if B consists only of nodes in G; otherwise, we say that B is *external*, in which case B may consist of both nodes in G and nodes not in G but in M.

2.2 Path Definitions

Consider a simple-area grid-like graph G. The input for a path selection problem is a set of N sources and destinations (i.e. packets), $\Pi = \{s_i, t_i\}_{i=1}^N$ in G. The output is a set of paths in G, $P = \{p_i\}_{i=1}^N$, 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 distance from s to t (the length of the shortest path from s to t) by $\mathsf{dist}(s, t)$. 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, $\mathsf{stretch}(P) = \max_i \mathsf{stretch}(p_i)$. We will denote by C the network congestion, which is the maximum number of paths in P that use any edge in the network. We will denote by C^* the optimal congestion incurred by the optimal set of paths the can route the packets in Π.

2.3 Cut Number

Consider a source s and destination t, both in G, and any other node v in G. We now define the *cut number* of node v with respect to s, t, denoted $cut_{s,t}(v)$. Let Q be any connected set of nodes containing v and not both of s, t (Q could contain one or none of s, t). The set Q is an $s - t$ node-cut if every path from s to t uses at least one node in Q. We will say that Q is an $s - t$ node cut for

node v. The *cut number* $cut_{s,t}(v)$ is the size (number of nodes) of the smallest $s-t$ node cut for node v.

A shortest path from s to v not containing t or from v to t not containing s give trivial node cuts. Note that at least one (but not necessarily both) of the trivial node cuts above must exist for every node v. Thus we have the following simple lemma,

Lemma 1. $1 \leq cut_{s,t}(v) \leq 1 + \min\{\mathsf{dist}_G(s,v), \mathsf{dist}_G(t,v)\}$.

Intuitively, nodes for which $cut_{s,t}(v) < 1 + \min\{\mathsf{dist}_G(s,i), \mathsf{dist}_G(t,i)\}$ represent bottlenecks in the network (with respect to sending packets from s to t). Consider two nodes u, v. Any node set containing v can be converted to one containing u by including the nodes in a shortest path from u to v. Thus, we have:

Lemma 2. $|cut_{s,t}(u) - cut_{s,t}(v)| \leq \mathsf{dist}_G(u,v)$.

The next lemma will be useful for our later results. It basically states that if a shortest path has to wind a lot, then this path must be passing through a small cut.

Lemma 3. *Suppose that p is a shortest path from s to t which crosses a vertical (or horizontal) line of nodes three times at the vertical (or horizontal) positions $x_1 > x_2 > x_3$ corresponding to the nodes u, v, w. Then $cut_{s,t}(v) < |x_1 - x_3|$.*

3 Square Sequences of Paths

Consider a simple-area grid-like graph G defined in the $m \times m$ grid M. Let p be a shortest path in G from a node s to node t. We define a *square sequence of shortest path* p, to be a sequence $R(p) = M_1, M_2, \ldots, M_k$ $(k \leq |p|)$ with the following properties:

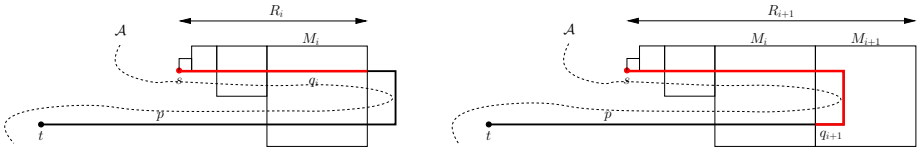
- *Canonical-Property:* each M_i is a canonical square (n_i and ℓ_i are the number of nodes and edges, respectively, in a side of M_i), and any two consecutive squares M_i and M_{i+1} are adjacent;
- *Coverage-Property:* p can be written as a concatenation of subpaths $p = p_1 p_2 \cdots p_k$, such that p_i is completely contained in M_i (subpath p_i is from node $v_i \in M_i$ to $w_i \in M_i$);
- *Cut-Property:* $\beta_1 \cdot \ell_i \leq cut_{s,t}(v_i) \leq \beta_2 \cdot n_i$, where $\beta_1, \beta_2 > 0$ are appropriately chosen constants.

The cut-property controls the cut numbers of the nodes in the square sequence of a shortest path. An effect of the cut-property is that adjacent squares do not differ significantly in size. Note that the canonical squares used in $R(p)$ may be internal of external. The same canonical external square may be used multiple times in a square sequence, or with two non-consecutive squares one may include the other, due to the way that the graph is formed. In section 3.3, we describe how to convert a square sequence to consist only of internal canonical squares, but at the expense of not covering all the nodes in the shortest path.

3.1 Square Sequence Construction

Given a shortest path p in G from a node s to node t we describe how to construct square sequence $R(p)$ recursively. We use the following notations: prefix subsequence $R_i(p) = M_1, \ldots, M_i$, prefix subpath $q_i = p_1 \cdots p_i$, and v_i and w_i denote the first and last nodes of p_i, respectively.

At the basis of the recursion we have $R_1(p) = M_1$, which is the canonical 1×1 square that contains s. Suppose that we have constructed the sequence $R_i(p)$. Let v_{i+1} be the first node after w_i in p. Select M_{i+1} to be the *largest* canonical square such that: M_{i+1} contains v_{i+1}, M_{i+1} is adjacent to M_i, and $\beta_1 \cdot \ell_{i+1} \leq cut_{s,t}(v_{i+1})$. The newly selected square M_{i+1} defines the subpath p_{i+1}. The process repeats until the destination node t is included. The following figures illustrate this process.



Notice in the above example, M_{i+2} may equal or contain M_i. Thus, the same submesh may be repeated in the canonical decomposition.

3.2 Side Size Bound

In what follows we will focus on providing a bound on the sum of the total side sizes $\sum_{i=1}^{n} n_i$ of the canonical squares in $R(p)$, with respect to the length $|p|$ of the shortest path p. This is useful because it help to bound the stretch of the resulting paths in the oblivious algorithm.

The connection of the square side sizes with the length of the path p comes from the cut-number relations of the squares. In the construction of the canonical squares, the side length ℓ_i is related to the cut number of v_i. We need to find a relation between the $|p_i|$ (the path segment in M_i) and the side length ℓ_i. We will establish this relationship through a sequence of lemmas.

The length of path segment p_i may exceed $2\ell_i$ in case that M_i is external. The first lemma shows that if a path segment is long enough then it has to cross some horizontal or vertical line three times.

Lemma 4. *If $|p_i| \geq 4\ell_i + 2$, then p_i must cross some horizontal or vertical line within M_i at least three times.*

The following result follows from Lemmas 3 and 4 and the cut-property of the square sequence:

Lemma 5. $|p_i| \leq 5\ell_i$.

Next we show that the sequence of canonical squares does not grow or decrease in size too rapidly.

Lemma 6. *$\beta_3 \cdot n_i \leq n_{i+1} \leq \beta_4 \cdot n_i$, for $1 \leq i \leq k-1$, and constants $0 < \beta_3 < 1/2 < \beta_4$.*

We continue with another result with respect to the length of the subpaths p_i inside some M_i.

Lemma 7. *Any segment q' of path p_i starting at v_i of length $|q'| = \alpha \leq \frac{\beta_1 \ell_i}{2}$ cannot cross a horizontal or vertical line 3 times.*

Proof. Suppose that q' intersects a line 3 times. By Lemma 3, there is node $u \in q'$ with cut number $cut_{s,t}(u) < \alpha$, and so $cut_{s,t}(v_i) < 2\alpha$ by Lemma 2. By construction of M_i, $\beta_1 \cdot \ell_i \leq cut_{s,t}(v_i) < 2\alpha \leq \beta_1 \cdot \ell_i$, which is a contradiction.

The next simple observation will be useful.

Lemma 8. *A path of length less than α can use at most 4 different $\alpha \times \alpha$ squares in the α-partition of M. Further all the different squares used lie within a $2\alpha \times 2\alpha$ square.*

Every time the path p moves from one of its canonical squares M_i to the next square M_{i+1}, we will say that *the path makes a canonical step*. Our next lemma basically states that if a path makes many (more than 4) canonical steps, then the path makes significant progress (relative to the length of the canonical squares).

Lemma 9. *If a segment q' of the shortest path p starting at v_i in M_i makes 5 canonical steps, then the length of that segment is at least $|q'| \geq \beta_5 n_i$, for a constant $\beta_5 > 0$.*

Proof. Let $\beta_5 = \beta_3^5$. Suppose that $|q'| < \beta_3^5 n_i$. Note that $|q'| \geq 4$, since q' makes five canonical steps. Since $\beta_3 < 1/2$, it has to be that $n_i > 2^7$. By Lemma 6, each canonical step can decrease the side length by at most a factor β_3. Let n_{min} be the minimum side size (number of nodes) of the canonical squares used over the next 5 canonical steps, then $n_{min} \geq \beta_3^5 n_i > |q'|$. Now consider the n_{min}-partitioning of M. Since $|q'| < n_{min}$, by Lemma 8, q' may only use at most 4 different $n_{min} \times n_{min}$ squares, all enclosed within a single $2n_{min} \times 2n_{min}$ square S. Each canonical transition must cross either the middle horizontal or middle vertical line of S. Since q makes 5 canonical transitions, either the middle horizontal or middle vertical line is crossed at least 3 times. To conclude, since $\beta_3 < 1/2$ and taking $\beta_1 > 2$, we get $|q'| < \beta_3^5 n_i < n_i - 1 = \ell_i < \frac{\beta_1 \ell_i}{2}$, which contradicts Lemma 7.

The next simple lemma follows directly from Lemma 1 and the fact that $(n_i - 1)\beta_1 = \ell_i \beta_1 \leq cut_{s,t}(v_i) \leq 1 + \mathsf{dist}(s,t)$.

Lemma 10. *$n_i \leq \beta_6 \cdot \mathsf{dist}(s,t)$, for some constant $\beta_6 > 0$.*

We are now ready to bound the sum of the sides of the squares in the canonical square decomposition of a shortest $s-t$ path p.

Theorem 1. *For the square sequence of shortest path p, $R(p) = M_1, \ldots, M_k$, the sum of the side sizes of the canonical squares is bounded as $\sum_{i=1}^{k} n_i \leq \beta_7 \cdot \mathsf{dist}(s,t)$, for some constant $\beta_7 > 0$.*

Proof. From Lemma 9, $|p_i| + |p_{i+1}| + |p_{i+2}| + |p_{i+3}| + |p_{i+4}| + |p_{i+5}| \geq \beta_5 n_i$, for $1 \leq i \leq k-5$. Thus,

$$\begin{aligned}\beta_5 \sum_{i=1}^{k-5} n_i &\leq \sum_{i=1}^{k-5} (|q_i| + |q_{i+1}| + |q_{i+2}| + |q_{i+3}| + |q_{i+4}| + |q_{i+5}|), \\ &\leq 6 \cdot \mathsf{dist}(s,t).\end{aligned}$$

By Lemma 10, $\sum_{i=k-4}^{k} n_i \leq 5\beta_6 \cdot \mathsf{dist}(s,t)$, and so by combining these two inequalities, we conclude that $\sum_{i=1}^{k} n_i \leq (6/\beta_5 + 5\beta_6) \cdot \mathsf{dist}(s,t)$.

3.3 Internal Square Sequence

The problem with the canonical decomposition $R(p)$ discussed in the previous section is that it constructs a sequence of squares which may not be completely enclosed in the network G. Here, we construct a sequence with only internal squares.

We sumarize the properties of the canonical square decomposition which will be important for the oblivious path selection algorithm.

1. The cut number of the nodes in the canonical square are proportional to the side of the canonical square.
2. The canonical squares in the sequence do not grow or shrink in size too quickly.
3. The sum of the side sizes of the canonical squares is proportional to the shortest path length $\mathsf{dist}_G(s,t)$.

We briefly sketch why these properties are important. The main idea is that we will construct the final path from random path segments which move from one canonical square to the next. The congestion caused by the paths within any one canonical square is related to the size of the square (how much the packets can spread). The cut number of these nodes gives a lower bound on the congestion for sending packets from s to t. Since the cut number and the side lengths are proportional (property 1 above), this allows us to show that the congestion inside the canonical squares is near optimal. Further, we do not want to create a bottleneck in going from one canonical square to the next, so there should be significant overlap between the intersecting sides of consecutive squares in the decomposition. This is ensured by property 2. Finally, property 3 bounds the stretch, since the path length is proportional to the sum of the side sizes of the canonical squares in the decomposition.

We now show how to convert the canonical square decomposition which may contain some external squares into an internal canonical square decomposition that still satisfies these three required properties. The basic idea is to replace

an external square with a sequence of one or more internal squares which are adjacent to it. This may result to a new sequence of squares which may not contain the shortest path p. However, this does not cause a problem since the goal of the oblivious algorithm is to construct new oblivious paths which are formed near the original path p, and not necessarily exactly on top of it.

Let $R(p) = M_1, M_2, \ldots, M_k$ be the canonical square decomposition for p, a shortest s-t path. We will construct an alternative path p' from s to t and a respective square sequence $\overline{R}(s,t) = M'_1, M'_2, \ldots, M'_{k'}$, such that all canonical squares in $\overline{R}(s,t)$ are internal. Similar to $R(p)$, let $\overline{R}_i(s,t)$ be a prefix of the square sequence $\overline{R}(s,t)$, q'_i a prefix of p', and p'_i the subpath of p' in M'_i. The path p' will be very close to p, so that $\mathsf{stretch}(p')$ is constant.

We know that M_1 is internal, by construction of $R(p)$. Let M_j, $j > 1$, be the first external canonical square in $R(p)$. We take the two square sequences to be the same up to M_{j-1}, namely, $\overline{R}_{j-1}(s,t) = R_{j-1}(p)$. The idea is that we will divert the path p from w_{j-1} (the last node in the canonical square before M_j) to v_{j+1} (the first node in the canonical square after M_j) so that the new respective prefix of path p' uses only internal canonical squares. Let $n_{min} = \min(n_{j-1}, n_j, n_{j+1})$ and consider the α-partition of M into squares of side $\alpha = n_{min}/2$. Assume for now that $n_{min} > 1$, and that M_{j-1}, M_j, M_{j+1} are disjoint. Note that M_j is also partitioned into squares of side α, as is M_{j-1} and M_{j+1}.

We now consider the (possibly partial) ring X of $\alpha \times \alpha$ squares in the α-partition of M which are adjacent to M_j but are not contained in M_{j+1}. The main claim, is that there is a path from w_{j-1} to v_{j+1} which uses a sequence of internal squares in the partial ring X, and then enters M_{j+1}. In particular, we consider the paths p_c and p_a from w_{k-1} to M_{j+1} which go around M_j, staying as close as possible M_j. The path p_c goes around M_j clockwise and the path p_a goes around M_j counter-clockwise. The following lemma is crucial in the construction:

Lemma 11. *Either p_a or p_c use exclusively internal squares in X.*

Proof. Suppose that both p_a and p_c use each at least one external square in X. Then, we will show that $cut_{s.t}(v_j) \leq 8n_j$. Let S_a be the first canonical square used by p_a which is external and let v_a be an external node in S_a. Similarly let S_c be the first canonical square used by p_c which is external and let v_c be an external node in S_c. Consider a shortest path from v_j to v_a and let Y_a be the segment of this path up to but excluding the first external node met on the path. Similarly let Y_c be the segment of a shortest path from v_j to v_c up to the first external node on the path. Let $Y = Y_a \cup Y_c$. Since the shortest path from v_j to v_a is entirely in a square of side $n_j + 2\alpha \leq 2n_j$ (since $2\alpha \leq n_j$), $|Y_a| \leq 4n_j$. Similarly, $|Y_c| \leq 4n_j$ and so $|Y| \leq 8n_j$. Clearly, Y is an s-t cut for v_j, since M_{j-1}, M_j, M_{J+1} are taken to be disjoint. Therefore, $cut_{s,t}(v_j) \leq 8n_j$. Since by construction of $R(p)$, $\beta_1 \ell_j \leq cut_{s,t}(v_j)$, and also $n_j = \ell_j + 1$ and we assumed that $n_j \geq 2$, by choosing the constant β_1 appropriately large, we obtain the $cut_{s,t}(v_i) > 8n_j$. A contradiction.

Lemma 11 implies that we can replace p_j with either p_a or p_c (whichever uses only internal $\alpha \times \alpha$ squares) in p'. The corresponding sequence $\overline{R}(s,t)$ is augmented after $\overline{R}_{j-1}(s,t)$ with the $\alpha \times \alpha$ internal canonical squares in ring X that correspond to the chosen p_a or p_c. This way, M_j has been eliminated and replaced by internal squares whose side size is within constant factor from n_j. The process repeats with the next external square in $R(p)$, until all external squares have been eliminated. The process can also be appropriately modified for the cases where $n_{\min} = 1$ and M_{j-1}, M_j, M_{j+1} are not pairwise disjoint. Using Theorem 1 and Lemma 6, we can obtain the following result:

Theorem 2. *Given source s and destination t we can construct a square sequence $\overline{R}(s,t) = M_1, M_2, \ldots, M_k$ such that:*

(i) each M_i is an internal canonical square (with side size n_i and length ℓ_i),
(ii) for any $v \in M_i$, $\gamma_1 \cdot \ell_i \leq cut_{s,t}(v) \leq \gamma_2 \cdot n_i$, for constants $\gamma_1, \gamma_2 > 0$,
(iii) M_i and M_{i+1} are adjacent with $\gamma_3 \cdot n_i \leq n_{i+1} \leq \gamma_4 \cdot n_i$, for $1 \leq i \leq k-1$, and constants $\gamma_3, \gamma_4 > 0$, and
(iv) $\sum_{i=1}^{k} n_i = \Theta(\mathsf{dist}(s,t))$.

4 Oblivious Path Selection Algorithm

Consider a hole-free grid-like network G with n nodes. The input for a path selection problem is a set of N sources and destinations (i.e. packets), $\Pi = \{s_i, t_i\}_{i=1}^{N}$ and the output is a set of paths, $P = \{p_i\}_{i=1}^{N}$, where each path $p_i \in P$ is from node s_i to node t_i. It suffices to describe the algorithm for an arbitrary single s-t source destination pair, and then each packet can use the same algorithm.

We will assume that in the network we have pre-computed the internal square sequence $\overline{R}(s,t) = M_1, M_2, \ldots, M_k$, for every pair s, t, as specified by Theorem 2. A *one-bend path* uses two straight lines in different dimensions, and a *two-bend path* uses three straight lines with alternate dimensions. The path from s to t is computed by using either a one-bend path or a two-bend in each square M_i. The decision for which type of path to use depends on the way that consecutive squares M_{i-1}, M_i, M_{i+1} are aligned. We have the following algorithm.

1: Consider three adjacent squares M_{i-1}, M_i, M_{i+1}. Let $n_{\min} = \min\{n_i, n_{i+1}\}$. Suppose we have constructed the path from s to t up to square M_{i-1}, and let u_{i-1} be the last node of the path on the side of M_{i-1} adjacent to M_i.
2: **if** (M_{i-1} and M_i are adjacent in the right and left sides, respectively, and M_i and M_{i+1} are adjacent in the top and bottom sides, respectively) **then**
3: Select a random node u_i among the $n_{\min}$ nodes on the top edge of M_i adjacent M_{i+1}. Construct a one-bend path from u_{i-1} to u_i.
4: **else**
5: **if** (M_{i-1} and M_i are adjacent in the right and left sides, respectively, and M_i and M_{i+1} are adjacent in the right and left sides, respectively) **then**

6: Select a random node $x \in M_i$ among n_i nodes on the horizontal line specified by u_{i-1}. Select a random node u_i among the $n_{\min}$ on the right edge of M_i adjacent to M_{i+1}. Construct a two-bend path from u_{i-1} to u_i through x.

7: Every other arrangement of the sides of M_{i-1}, M_i, M_{i+1} can be handled similar to one of the cases above. For the first square we have $u_1 = s$ and for the last square $u_k = t$.

5 Congestion and Stretch Analysis

From the algorithm description in Section 4, the path from s_i to t_i is formed by concatenating one-bend paths or two-bend paths formed in the respective squares in the seuquence $\overline{R}(s_i, t_i)$. By construction, we observe that each subpath in a square M_i has length at most $3n_i$. This observation with combination of property (iv) in Theorem 2 implies that:

Theorem 3 (Stretch). *The stretch of the paths P returned by the oblivious algorithm has* $\mathsf{stretch}(P) = \Theta(1)$.

Thus, we only need to focus on the congestion. We start with a lower bound analysis for the optimal congestion and then we give an upper bound which is within a log factor from the lower bound. In what follows we give the bounds in terms of the *node congestion*, which is the maximum number of paths that use any node. The resulting bounds immediately translate to edge congestion within a factor 4 since each node has at most four adjacent edges. Thus, the asymptotic bounds stay the same for edge congestion.

5.1 Lower Bound on Optimal Congestion

Consider an arbitrary $\alpha \times \alpha$ internal canonical square B. Let $Q \subseteq P$ be the set of paths selected by the algorithm that use B because it is in their square sequence. We establish a lower bound on the optimal congestion C^* due to the paths Q in B.

With respect to any path $p \in Q$ from source s to destination t, Theorem 2 implies that every node $v \in B$ has $cut_{s,t}(v) \leq \gamma_2\alpha$. Let H be a $(2\gamma_2+1)\alpha \times (2\gamma_2+1)\alpha$ grid subgraph of nodes in M such that B is in the middle of H. We truncate H wherever it exceeds the network M, resulting to a rectangular grid of maximum side length $(2\gamma_2+1)\alpha$. Let H_G denote the subgraph of G in H. Note that H_G may be disconnected. We define the *perimeter* nodes of H_G, denoted $T(H_G)$, as the set of nodes of $u \in H_G$ which have incident edges $(u, v) \in G$, such that $v \notin H_G$.

Lemma 12. *The following properties hold for $T(H_G)$: (i) $|T(H_G)| \leq (8\gamma_2+4)\alpha$, and (ii) for any $w_1 \in B$ and $w_2 \in T(H_G)$, $\mathsf{dist}(w_1, w_2) \geq \gamma_2\alpha$.*

Proof. We define the perimeter nodes of H, and we denote them as $T(H)$, to be those nodes of H which have incident edges $(u, v) \in M$, where $u \in M$ and $v \notin M$. Since H is a rectangular grid of maximum side size 3α, $|T(H)| \leq 4 \cdot (2\gamma_2+1)\alpha = (8\gamma_2 + 4)\alpha$. We will show that $T(H_G) \subseteq T(H)$.

Let $u \in T(H_G)$, and suppose that $u \notin T(H)$. Then there is an edge $(u, v) \in G$, with $v \notin H_G$. Since $u \in H$ and $u \notin T(H)$, we obtain $v \in H$. Thus, v is a node in H and also a node in G, and therefore, by construction of H_G, $v \in H_G$, a contradiction.

Therefore, $T(H_G) \subseteq T(H)$. Consequently, $|T(H_G)| \leq |T(H)| \leq (8\gamma_2 + 4)\alpha$, which proves property (i). Further, since B is chosen to be in the middle of H (before the truncation of H), the smallest distance of any node $w_1 \in B$ to the closest perimeter node of H is at least $\gamma_2\alpha$. Since $T(H_G) \subseteq T(H)$, w_1 is at distance at least $\gamma_2\alpha$ from any node $w_2 \in T(H_G)$, proving property (ii).

We can write $Q = Q_1 \cup Q_2$, where Q_1 are the paths of Q whose both source and destination are outside H_G, and Q_2 are the paths of Q whose either source or destination (or both) are inside H. Note that Q_1 and Q_2 are disjoint. We first relate Q_1 with the lower bound C^*.

Lemma 13. *Given a path $p \in Q_1$ with source s and destination t, every path from s to t uses some node in H_G.*

Proof. Suppose that there is a path $q \in G$ from s to t such that for each node $u \in q$ with $u \notin H_G$. Let $v \in B$. Consider now the smallest (s, t)-cut z, such that $v \in z$. Clearly, $cut_{s,t}(v) = |z|$, where $|z|$ denotes the number of nodes in z. Clearly, z contains a node $u \in q$. Since, $v \in B$ and $u \notin H_G$, z has to use a node in the perimeter $w \in T(H_G)$. Thus, z has two edge-disjoint sets z_1 and z_2, from v to w, and from w to u, respectively.

From Lemma 12, $|z_1| \geq \gamma_2\alpha + 1$. Further, $|z_2| \geq 1$, since $u \neq w$ ($w \in T(H_G)$ and $u \notin T(H_G)$). Consequently, $|z| \geq |z_1| + |z_2| - 1 \geq \gamma_2\alpha + 1$. Therefore, $cut_{s,t}(v) \geq \gamma_2\alpha + 1$. A contradiction, since by Theorem 2, $cut_{s,t}(v) \leq \gamma_2\alpha$.

Lemma 14. *$C^* \geq \gamma_5|Q_1|/\alpha$, for some constant $\gamma_5 > 0$.*

Proof. Consider a path $p \in Q_1$ from s to t. From Lemma 13, each path q from s to t has to use a node in H_G. Since $s \notin H_G$ and $t \notin H_G$, q enters H_G through one of the perimeter nodes in $T(H_G)$.

Thus, any path selection algorithm (including the optimal oblivious or non-oblivious) for the $|Q_1|$ source-destination pairs, has to construct paths that each uses at least one node in $T(H_G)$. Let $U_{total} = |Q_1|$ denote the total node utilization of the nodes in $T(H_G)$ due to the $|Q_1|$ paths that have to be constructed. The average node utilization of the nodes in $T(H_G)$ is $U_{avg} = U_{total}/|T(H_G)|$. The optimal congestion C^* has to be at least as much as the average node utilization U_{avg}, since some node in $T(H_G)$ has to be used by at least U_{avg} paths, by any path selection algorithm. Thus, Lemma 12 gives for a constant γ_5:

$$C^* \geq U_{avg} \geq \frac{U_{total}}{|T(H_G)|} \geq \frac{|Q_1|}{(8\gamma_2 + 4)\alpha} = \frac{\gamma_5|Q_1|}{\alpha}.$$

We continue now to relate the optimal congestion C^* to the paths in Q_2.

Lemma 15. *Let a path $p \in Q_2$ from s to t. Then, $\mathsf{dist}(s,t) \geq \gamma_6(\alpha-2)$, for some constant $\gamma_6 > 0$.*

Proof. Let q be a shortest-path from s to t which is used to construct the sequence of squares $R(q)$ from s to t containing possibly external or internal squares. Path p uses B because it appears in the sequence of internal squares $\overline{R}(s,t)$ that we obtain from $R(q)$. We examine two cases:

- $B \in R(q)$: Then, shortest path q uses some node $v \in B$. From Theorem 2, $\gamma_1(\alpha-1) \leq cut_{s,t}(v)$. From Lemma 1, $|q| \geq cut_{s,t}(v) - 1 \geq \gamma_1(\alpha-1) - 1 \geq \gamma_1(\alpha-2)$.
- $B \notin R(q)$: By the construction of $\overline{R}(s,t)$, B must be adjacent to some $\alpha' \times \alpha'$ canonical square $B' \in R$, where $\alpha' \geq 2\alpha$, such that q goes through B'. Further, if v is the first node of q that uses B', then there is a node $u \in B$ such that $\mathsf{dist}(u,v) \leq \kappa\alpha$, for some appropriate constant κ. From Theorem 2, $cut_{s,t}(u) \geq \gamma_1(a-1)$. From Lemma 2, $cut_{s,t}(v) \geq cut_{s,t}(u) - \mathsf{dist}(u,w) \geq \gamma_1(\alpha-1) - \kappa\alpha$ From Lemma 1, $|q| \geq cut_{s,t}(v) - 1 \geq (\gamma_1 - \kappa)(\alpha-2)$.

Considering now both of the cases, $|q| \geq \min\{(\gamma_1-\kappa)\alpha, \gamma_1(\alpha-2)\} = \gamma_6(\alpha-2)$, for some appropriate constant γ_6.

We define the k-neighborhood of a node in graph G as $N_k(v) = \{u \in G : \mathsf{dist}(u,v) \leq k\}$ (note that this includes also v since by default $dist(v,v) = 0$). The k-neighborhood of a set of nodes A in G is $N_k(A) = \bigcup_{v \in A} N_k(v)$.

Lemma 16. *$C^* \geq \gamma_7 |Q_2|/\alpha$, for some constant $\gamma_7 > 0$.*

Proof. Consider the case where $\alpha \geq 4$. Let $A = N_{\gamma_6\alpha}(H_G)$, that is, A is the $\gamma_6\alpha$-neighborhood of H_G in G (note that A includes all the nodes in H_G). Let H' be a square grid of side length $2^{\lceil \log(2\gamma_6\alpha + (2\gamma_2+1)\alpha)\rceil} = \xi\alpha$, for some constant ξ, such that B is in the middle. (Note that the non truncated square grid H with side length $(2\gamma_2+1)\alpha$ is also in the middle of H'.) In particular, consider the truncated version (due to the boundaries of M) of H' which is a rectangular grid of maximum side length $\xi'\alpha$ that contains H and B. We have that all the nodes in A are inside H'. Therefore, $|A| \leq (\xi\alpha)^2$.

Consider now a path $p \in Q_2$ from s and t. Suppose, without loss of generality, that $s \in H_G$ (the other case is $t \in H_G$ which is symmetric). By Lemma 15, any path p' from s to t has $|p'| \geq \gamma_6(\alpha-2)$. Since the source of p' is in H_G, path p' has a prefix p'' which is completely inside A and has $|p''| = \gamma_6(\alpha-2)$. Thus, p' uses at least $\gamma_6(\alpha-2)$ nodes of A. Similarly, any path from s to t has to use at least $\gamma_6(\alpha-2)$ nodes of A.

Thus, any path selection algorithm for the $|Q_2|$ source-destination pairs, has to construct paths that each uses at least $\gamma_6(\alpha-2)$ nodes in $T(H_G)$. Let $U_{total} = \gamma_6(\alpha-2)|Q_2|$ denote the total node utilization of the nodes in A due to the $|Q_2|$ paths that have to be constructed. The average node utilization of the nodes in A is $U_{avg} = U_{total}/|A|$. The optimal congestion C^* has to be at least as much

as the average node utilization U_{avg}, since some node in A has to be used by at least U_{avg} paths, by any path selection algorithm. Since we took $\alpha \geq 4$, we obtain:

$$C^* \geq U_{avg} \geq \frac{U_{total}}{|A|} \geq \frac{\gamma_6(\alpha-2)|Q_2|}{(\xi\alpha)^2} \geq \frac{\gamma_7|Q_2|}{\alpha},$$

for some constant γ_7. For the case $\alpha < 4$, we can use the trivial bound $C^* > 1$, and adjust appropriately γ_7.

Lemma 17. *$C^* \geq \gamma_8|Q|/\alpha$, for some constant $\gamma_8 > 0$.*

Proof. By combining Lemma 14 and Lemma 16, we obtain $C^* \geq \max\{\gamma_5|Q_1|/\alpha, \gamma_7|Q_2|/\alpha\}$. Let $X = \max\{|Q_1|, |Q_2|\}$. Since Q_1 and Q_2 are disjoint, $X \geq |Q|/2$. Let $\gamma_{\min} = \min\{\gamma_5, \gamma_7\}$, and $\gamma_8 = \gamma_{\min}/2$. We have,

$$C^* \geq \max\left\{\frac{\gamma_{\min}|Q_1|}{\alpha}, \frac{\gamma_{\min}|Q_2|}{\alpha}\right\} \geq \frac{\gamma_{\min}X}{\alpha} \geq \frac{\gamma_{\min}|Q|}{2\alpha} = \frac{\gamma_8|Q|}{\alpha}.$$

5.2 Upper Bound on Algorithm Congestion

We continue with providing an upper bound on the congestion. Let $Q \subseteq P$ be the set of chosen paths by the algorithm that use internal canonical $\alpha \times \alpha$ square B because it is in their internal square sequence.

Lemma 18. *For any $v \in B$, the number of paths in Q which are expected to use v is at most $\varphi|Q|/a$, for some constant φ.*

Proof. We can write $Q = Q_a \cup Q_b$, where Q_a are the chosen paths that follow one-bend paths in B, while Q_b are the chosen paths that follow two-bend paths in B. Note that Q_a and Q_b are disjoint.

Consider a path $p \in Q_a$. Let p' be the one-bend subpath of p in B. We can write p' as the concatenation of two paths $p' = p_1p_2$, where p_1 corresponds to the first part of p' before the bend (the bend node is also included in p_1), and p_2 corresponds to the second part of p' after the bend. Path p may use v either in one of the subpaths p_1 or p_2. Suppose, without loss of generality, that p_1 is horizontal. Let u be the first node in p_1. If u is in the same row with v then p_1 uses v. According to the path selection algorithm, node u is chosen (along a column of nodes) with probability at most $r = \varphi/\alpha$, for some constant $\varphi \geq 1$. Therefore, subpath p_1 uses v with probability at most r. With a similar analysis, subpath p_2 uses v with probability at most r (by adjusting appropriately φ). Therefore, path p uses v with probability at most r.

Consider now the case where $p \in Q_b$. Let p' be the two-bend subpath of p in B. We can write p' as the concatenation of three paths $p' = p_1p_2p_3$, which correspond to the part of p' before the first bend, between the first and second bends, and after the second bend, respectively. Path p may use v in only one of the subpaths p_1 or p_2 or p_3. The probability that either p_1 or p_3 use v is bounded by r, as proven in the one-bend case. Suppose, without loss of generality, that p_2 is horizontal. Let u be the first node of p_2. If u is in the same row with v then

p_2 uses v. According to the path selection algorithm, node u is chosen (along a column of nodes) with probability $r' = 1/\alpha$. Therefore, subpath p_2 uses v with probability at most r'. Therefore, path p uses v with probability at most $\max(r, r') = r$.

Therefore, from the paths in Q_a it is expected that at most $|Q_a|r$ will use v. Similarly, the expected number of paths from Q_b that will use v is at most $|Q_b|r$. Therefore, the expected number of paths from Q that will use v is at most $|Q_b|r + |Q_b|r = |Q|r = \varphi|Q|/a$.

From Lemma 17 and Lemma 18, we obtain the following corollary.

Corollary 1. *For any $v \in B$, the number of paths in P which are expected to use v is $O(C^*)$.*

Theorem 4 (Congestion). *The expected congestion on any node $v \in G$ is $O(C^* \log n)$, where n is the number of nodes in G. (The same result holds also with high probability by applying a Chernoff bound.)*

Proof. Node $v \in G$ may participate to $O(\log m)$ partition levels of internal canonical squares. Since we can choose M to be such that m is at most the diameter D of G, then we have that $m = \Theta(D) = \Theta(n)$, which implies $O(\log n)$ partition levels. From Corollary 1, the internal square at any particular partition level causes $O(C^*)$ expected congestion to v. Thus, the total expected congestion to v is $O(C^* \log n)$.

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. 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.
10. 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.
11. 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.
12. 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.
13. 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.
14. Lucian Popa, Afshin Rostamizadeh, Richard Karp, Christos Papadimitriou, and Ion Stoica. Balancing traffic load in wireless networks with curveball routing. In *MobiHoc*, 2007.
15. 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.
16. 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.
17. P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365–374, 1987.
18. C. Scheideler. Course notes. `http://www14.in.tum.de/lehre/2005WS/na/index.html.en`.
19. 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.
20. L. G. Valiant. A scheme for fast parallel communication. *SIAM Journal on Computing*, 11:350–361, 1982.
21. 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.
22. Feng Zhao and Leonidas J. Guibas. *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann, 2004.