Sparse Spanners vs. Compact Routing

Cyril Gavoille*
LaBRI - Université de Bordeaux
351, cours de la Libération
33405 Talence cedex, France
gavoille@labri.fr

Christian Sommer†
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139-4307
csom@mit.edu

ABSTRACT

Routing with *multiplicative* stretch 3 (which means that the path used by the routing scheme can be up to three times longer than a shortest path) can be done with routing tables of $\tilde{\Theta}(\sqrt{n})$ bits[1] per node. The space lower bound is due to the existence of dense graphs with large girth. Dense graphs can be sparsified to subgraphs, called *spanners*, with various stretch guarantees. There are spanners with *additive* stretch guarantees (some even have constant additive stretch) but only very few additive routing schemes are known.

In this paper, we give reasons why routing in unweighted graphs with *additive* stretch is difficult in the form of space lower bounds for general graphs and for planar graphs. We prove that any routing scheme using routing tables of size μ bits per node and addresses of poly-logarithmic length has additive stretch $\tilde{\Omega}(\sqrt{n/\mu})$ for general graphs, and $\tilde{\Omega}(\sqrt{n}/\mu)$ for planar graphs, respectively. Routing with tables of size $\tilde{O}(n^{1/3})$ thus requires a polynomial additive stretch of $\tilde{\Omega}(n^{1/3})$, whereas spanners with average degree $O(n^{1/3})$ and *constant* additive stretch exist for all graphs. Spanners, however sparse they are, do not tell us how to route. These bounds provide the first separation of sparse spanner problems and compact routing problems.

On the positive side, we give an almost tight upper bound: we present the first non-trivial compact routing scheme with $o(\lg^2 n)$-bit addresses, *additive* stretch $\tilde{O}(n^{1/3})$, and table size $\tilde{O}(n^{1/3})$ bits for all graphs with linear local tree-width such as planar, bounded-genus, and apex-minor-free graphs.

*C.G. is also member of "l'Institut Universitaire de France". He is also supported by ANR projects "ALADDIN", and the équipe-projet INRIA Sud-Ouest "CÉPAGE'.

†Part of this work was done at the University of Tokyo and while C.S. visited LaBRI.

[1]Tilde-big-O notation is similar to big-O notation up to factors poly-logarithmic in n.

SPAA'11, June 4–6, 2011, San Jose, California, USA.

Categories and Subject Descriptors

C.2.2 [**Network Protocols**]: Network Architecture and Design—*Routing protocols*; F.2.2 [**Nonnumerical Algorithms and Problems**]: [Routing and layout]; G.2.2 [**Discrete Mathematics**]: Graph Theory—*Network problems, Graph labeling, Graph algorithms*.

General Terms

Algorithms, Theory.

Keywords

compact routing, shortest paths

1. INTRODUCTION

Routing is essential for communication in networks. For a network with n devices, we deem a routing scheme to be *compact*, if the maximum amount of memory used per routing table grows asymptotically slower than n. Research on *compact routing* [8, 20, 34, 35, 52] is concerned with the tradeoff between the space requirements and the quality of the routes, where the quality is measured with respect to the best possible route. The (multiplicative) *stretch* of a routing scheme is defined as the worst-case ratio for any pair of nodes of the route length divided by the shortest-path distance. A routing scheme that uses linear space per node may store information on all shortest routes. If the memory available per node (the *table size*) is $o(n \lg n)$, packets can not always be sent along shortest paths [41].

The fundamental tradeoff between stretch and routing table size has been investigated broadly and both upper and lower bounds for general graphs [4, 7, 20, 55] and for graphs stemming from specific classes such as trees [31, 32, 47, 48, 55], planar graphs [37, 49, 54], minor-free graphs [1, 3], non-positively curved plane graphs [18], graphs with bounded genus [37], with low doubling dimension [2, 45], chordal graphs [23, 24], "flat" networks [43], random graphs [28], power-law graphs [13, 16], permutation graphs, interval graphs and related classes [11, 25, 26], and others are known (see Table 1 for an overview).

The space lower bounds known for general graphs [36, 55] are based on dense graphs with given *girth*.[2] To overcome difficulties with dense graphs, sparse *spanners* [6, 19, 51] have been devised. Spanners are subgraphs with fewer edges that satisfy certain distance inequalities — spanners

[2]The girth of a graph is the length of its shortest cycle.

Graph	Stretch	Tables	Addresses	Ref.
General	1	$n \lg_2 n$	$O(\lg n)$	Folkl.
General	2	$(n-\sqrt{n}) \lg_2 n$	$O(\lg n)$	[42]
General	3	$\tilde{O}(n^{1/2})$	$O(\lg n)$	[55]
General	$4k-5$	$\tilde{O}(n^{1/k})$	$o(k \lg^2 n)$	[55]
Trees	1	none	$o(\lg^2 n)$	[31, 55]
Tree-width τ	1	none	$O(\tau \lg^2 n)$	[50]
Planar	1	$7.18n + o(n)$	$O(\lg n)$	[49]
Genus γ	1	$n \lg_2 \gamma + O(n)$	$O(\lg n)$	[37]
Planar	$1+\epsilon$	none	$o(\epsilon^{-1} \lg^2 n)$	[54]
Minor-free	$1+\epsilon$	none	$o(\epsilon^{-1} \lg^2 n)$	[1]
Doubling dim. α	$1+\epsilon$	$\epsilon^{-O(\alpha)} \lg^3 n$	$O(\lg n)$	[45]
$G_{n,p}$ $(p=1/2)$	1	$n + O(\lg^4 n)$	$O(\lg n)$	[39]
$G_{n,p}$ $(np \in (\lg n, \sqrt[9]{n}))$	2	$\tilde{O}(n^{3/4})$	$O(\lg n)$	[28]
Random PL	5	$\tilde{O}(n^{1/3})$	$o(\lg^2 n)$	[16]

Table 1: Best results known on labeled (name-dependent) compact routing schemes for connected graphs on n nodes. The stretch is *multiplicative*, meaning the worst-case ratio between the path used by the routing scheme and the distance between source and target. For labeled schemes, the designer may choose arbitrary node names (also termed *addresses*), including, for example, names that depend on the topology and the edge weights of the graph.

ought to maintain distances up to small *stretch* factors. Recently, instead of spanners with multiplicative stretch, *additive spanners* [10, 12, 27, 53, 57, 58, 59] have been investigated as well. Spanners could potentially be used for routing — in fact, their usefulness for routing is often one of the (main) reasons stated in the introduction and motivation section of articles on spanners. However, the tradeoff between routing table space requirements and worst-case stretch is not yet completely understood for sparse graphs. Indeed, sparse spanners do not tell the designer of the routing scheme how to find and encode short routes.

The *additive* stretch of a routing scheme (and, analogously, of a spanner) is defined as the worst-case difference for any pair of nodes of the route length minus the shortest-path distance (the graphs considered are assumed to be *unweighted* whenever we consider additive stretch). Instead of routing with multiplicative stretch, researchers have also started to investigate routing schemes with additive stretch guarantees. However, only very little is known (see Table 2 for an overview).

For general graphs, the following straightforward approach guarantees additive stretch β using routing tables of size $\tilde{O}(n/\beta)$, for any integral parameter β. The routing scheme routes along shortest-path spanning trees rooted at each node of a small $\frac{1}{2}\beta$-dominating set, that is a subset C of nodes such that every node u is at distance at most $\frac{1}{2}\beta$ from some *center* $c_u \in C$. It is well-known that every connected graph has a $\frac{1}{2}\beta$-dominating set of size $< 2n/(\beta+1)$, computable efficiently [46]. By the triangle inequality, routes are stretched by an additive factor of at most β. The address of each node u consists of the node identifier of its closest center c_u, telling the source which tree to use. Since each tree contributes $o(\lg^2 n)$ bits per node to the routing tables [31, 55], the tables are of size $o(|C| \lg^2 n) = \tilde{O}(n/\beta)$.

A non-trivial compact routing scheme with additive stretch β should thus have table size $o(n/\beta)$. Schemes with small tables and small additive stretch have been devised for restricted graph classes such as chordal graphs [24], graphs with bounded tree-length [23], and, more generally[3] graphs of bounded hyperbolicity [17]. Furthermore, Brady and Cowen [14] construct a routing scheme with additive stretch 6 given an *exact* distance labeling scheme [38, 40, 50]. Their approach yields sublinear table sizes for all graphs that allow for a distance labeling scheme with labels of length $o(\sqrt{n})$. Unfortunately, any exact distance labeling scheme in unweighted graphs requires at least $\Omega(n)$-bit labels in general, $\Omega(\sqrt{n})$-bit labels for bounded-degree graphs and $\Omega(n^{1/3})$-bit labels for planar graphs [40].

Other compact routing schemes have been proposed for internet-like topologies, with small multiplicative stretch and poly-logarithmic additive stretch [13, 33].

Graph	Stretch	Table	Addresses	Ref.
General	β	$\tilde{O}(n/\beta)$	$o(\lg^2 n)$	Folkl.
Diameter Δ	2Δ	none	$o(\lg^2 n)$	Folkl.
$\ell(n)$-Labels	6	$O(\sqrt{n}(\ell(n)+\lg^2 n)$	$O(\ell(n)+\lg^2 n)$	[14]
Interval	1	$O(\lg n)$	$O(\lg n)$	[14]
Circular-arc	1	$O(\lg n)$	$o(\lg^2 n)$	[14]
Chordal	2	$o(\lg^3 n)$	$o(\lg^3 n)$	[24]
Tree-length δ	$6\delta-2$	$O(\delta \lg^2 n)$	$O(\delta \lg^2 n)$	[23]
δ-Hyperbolic	$O(\delta \lg n)$	$O(\delta \lg^2 n)$	$O(\delta \lg^2 n)$	[17]

Table 2: Best results known on compact routing schemes with *additive* stretch for unweighted connected graphs on n nodes. The scheme by Brady and Cowen [14] uses exact *distance labels* of length $\ell(n)$ to devise a routing scheme with additive stretch.

Although there are some routing schemes that guarantee additive stretch for restricted classes of graphs, the results on routing somehow cannot catch up with the results on spanners. While there are rather sparse additive spanners, additive routing schemes for (more) general graphs have not been found.

1.1 Contributions

In the current work, we investigate tradeoffs between the size of routing tables and the additive stretch. We give both upper and lower bounds. Our lower bounds explain why, unfortunately, sparse additive spanners cannot be converted into compact routing schemes. Routing with additive stretch requires large tables — even for planar graphs. We thus give the first separation of spanner problems and routing problems. On the positive side, we also give an almost tight upper bound for routing with additive stretch on graphs with linear local tree-width (a class of graphs that includes planar, bounded-genus, and apex-minor-free graphs).

Lower Bounds.

For general graphs on n nodes, we prove that any routing scheme using addresses of poly-logarithmic length and routing tables of size μ bits per node has additive stretch at least $\tilde{\Omega}(\sqrt{n/\mu})$ (Theorem 1). For planar graphs, we prove that the lower bound on the additive stretch is at least $\tilde{\Omega}(\sqrt{n}/\mu)$ (Theorem 2).

[3]Chordal graphs have tree-length 1, graphs of tree-length δ are $O(\delta)$-hyperbolic, δ-hyperbolic graphs have tree-length $O(\delta \lg n)$.

Upper Bound.

We provide a rather general approach for compact routing with additive stretch. For planar graphs on n nodes (and actually for graphs of linear local tree-width, which includes all bounded-genus graphs), this general approach yields a compact routing scheme with poly-logarithmic addresses, table size $\tilde{O}(n^{1/3})$, and additive stretch $\tilde{O}(n^{1/3})$ (Theorem 3). Our upper bound is almost tight with respect to the lower bound of Theorem 2, which says that table size $\tilde{O}(n^{1/4})$ requires additive stretch $\tilde{\Omega}(n^{1/4})$. Actually, our scheme works for general graphs but with weaker bounds on the memory consumption.

2. LOWER BOUNDS

We first state and prove the general lower bound (Theorem 1). Second, we state and prove the lower bound for planar graphs (Theorem 2) as a special case of the general lower bound.

THEOREM 1. *For connected, unweighted graphs on n nodes, any labeled routing scheme using addresses of poly-logarithmic length and routing tables of size μ bits per node has worst-case additive stretch at least $\beta = \Omega(\sqrt{n/\mu})$.*

Before presenting our proof, let us observe that one cannot derive a lower bound on the additive stretch from multiplicative-stretch lower bounds for the following reason. It is known [36, 55] that there are worst-case dense graphs on k nodes for which any routing scheme with $o(k)$-bit routing tables cannot achieve shortest-path routing between all pairs of adjacent nodes (and so provide routes of length at least 2 for some edge uv). We may believe that by uniformly subdividing each edge of these dense graphs into k new edges we are done. Choosing $k \sim n^{1/3}$, we may obtain a graph with $O(k^2)$ edges and $O(k^3) = O(n)$ nodes in which any routing scheme between some distance-k nodes requires a route of length $\geq 2k$, and thus an additive stretch of $\geq 2k - k = n^{1/3}$. The argument is flawed since the route from u to v in the new graph can first make a short loop to collect routing information, before going straight to v. And, unfortunately, the degree of u is large by construction, and thus the routing information about u's neighbors could be efficiently distributed around u (e.g., by the use of some hash tables).

This reduction might work if worst-case graphs of bounded degree were known. Unfortunately, compact routing with small (multiplicative) stretch in bounded degree graphs is widely open. No upper and lower bounds (other than the ones for general graphs) are known for this problem.

PROOF. Our worst-case graphs (see Fig. 1) consist of a set of p sources $S = \{s_1, \dots s_p\}$, a set of q targets $T = \{t_1, \dots t_q\}$, and two graphs L (for left) and R (for right), both lying between the sources and the targets. Each source s_i is connected to a representative node s_i^L in L and a node s_i^R in R, on a path of length ρ, respectively. L and R connect these representative sources to the targets t_j. Both L and R are built from the same base graph (or *gadget*) but different shortcuts are added. For each pair (s_i, t_j) there is exactly one shortcut, *either* in L *or* in R. Except for these shortcuts, L and R thus look almost identical. For each pair (s_i, t_j), *independent of all other pairs*, there is only one shortest path, going *either* through L *or* through R (depending on whether the shortcut is in L or in R). The graphs are constructed such that any alternative path is much longer.

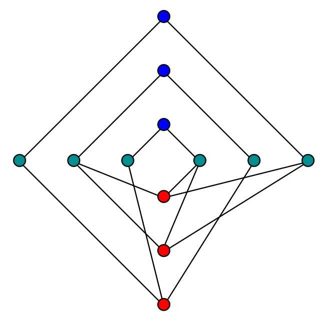

Figure 1: An example of the lower bound construction for general graphs. Each target (red, lower part) is connected to either the left part or the right part (independently for different targets). Intuitively speaking, to route from any source we need to know for all targets whether to send a message using the edge on the left or the edge on the right. If the number of sources is sufficiently large, the information cannot be encoded in the address.

Let K be the $p \times q$ matrix with entries in $\{0, 1\}$, where $k_{i,j}$ is 0 if the shortcut for (s_i, t_j) is in L and 1 otherwise (i.e., the shortcut is in R). Since each shortcut (i, j) can be added either to L or to R independently, there are 2^{pq} different combinations and thus 2^{pq} different matrices K. An encoding of K thus requires at least $\lg_2 2^{pq} = pq$ bits. In the following, we argue that the addresses of the targets and the routing tables "around" the sources must *encode* K.

Intuitively speaking, to route from s_i, we need to know for all t_j whether to send a message using L or R. For any source s_i, a routing scheme that has additive stretch at most 2ρ may explore the routing tables of all the nodes at distance at most ρ from s_i to decide whether to use L or R. Recall that, in our construction, each source is connected to only two long paths of length ρ and thus the number of nodes within distance ρ is 2ρ. The collective information of all the nodes (including s_i) within distance ρ around source s_i is bounded by $(2\rho + 1)\mu$ bits. The collective information of all the nodes within distance ρ around all the p sources is bounded by $p(2\rho + 1)\mu$ bits. The addresses of the p sources and the q targets may also contribute to the encoding of K, adding another $(p + q)\alpha$ bits, where α is the maximum address length. We thus have

$$p(2\rho + 1)\mu + (p + q)\mu \quad \geq \quad pq. \tag{1}$$

For the case of general graphs, the base graph (gadget) is very simple (Fig. 1 provides an illustration): it consists of p nodes $s_i^{\{L,R\}}$ (one for each source) and no edges. A "shortcut" for a pair of source and target (s_i, t_j) is a path of length ρ from $s_i^{\{L,R\}}$ to t_j. The graph uses $\Theta(p\rho)$ nodes and edges to connect S to L and R and $\Theta(pq\rho)$ nodes and edges to connect L and R to T. The shortest path from s_i to t_j has length 2ρ. Suppose that the shortcut (i, j) was in L. Any path from s_i to t_j in R has length at least $\rho + 3\rho$. To achieve additive stretch less than 2ρ, the routing scheme must know after ρ steps whether to use L or R.

Let us now fix the parameters p, q, ρ, with respect to α

(the address length), μ and n. The number of sources p is chosen such that the address of a target t_j cannot encode the L vs. R decision for each source s_i:

$$p = \omega(\alpha).$$

For poly-logarithmic address length α, we may choose p to be poly-logarithmic in n as well. Using $n = \Theta(pq\rho)$, Eq. (1) yields

$$\begin{aligned} p(2\rho+1)M + (p+q)\alpha &\geq pq \quad (2) \\ \rho M &= \tilde{\Omega}(q) \quad (3) \\ \rho &= \tilde{\Omega}(\sqrt{n/\mu}). \quad (4) \end{aligned}$$

Since the additive stretch is at least ρ, the claim follows. □

We observe that the worst-case graphs used in the our proof have less than $2n$ edges. The graphs itself are sparse spanners (trivial stretch). However, we prove that no additive compact routing scheme with small tables exists. Note that the optimal $\tilde{O}(n^{1/k})$-space routing scheme of Thorup and Zwick [55] also requires additive stretch no better than $n^{1/2-O(1/k)}$ for these graphs. However, the sampling technique used to design this optimal routing scheme has also been used [57] to produce spanners for unweighted graphs with stretch much smaller than $O(k)$, namely $1+\epsilon$ for any $\epsilon > 0$.

Note.

Techniques for compact routing schemes and for distance oracles [56] are often applicable to both problems. The graph (and the query pairs) we use in this lower bound, however, admits a straightforward, exact distance (and shortest-path) oracle using linear space. Our proof shows that the routing problem is much harder.

The lower-bound graph construction for general graphs can be combined with lower bounds for exact compact routing schemes [2]. Note that the upper part of the general construction is a planar graph. We thus need a planar gadget for the lower part. The general construction, combined with [2], yields a lower bound for additive stretch routing in planar graphs, and also bounded-doubling-dimension graphs (details and proof in Appendix A).

Theorem 2. *For connected, unweighted bounded degree planar graphs on n nodes, any labeled routing scheme using addresses of poly-logarithmic length and routing tables of size μ bits per node has worst-case additive stretch at least $\beta = \tilde{\Omega}(\sqrt{n}/\mu)$.*

3. UPPER BOUND

In this section, we provide a routing scheme with table sizes and additive stretch both $\tilde{O}(n^{1/3})$ for planar graphs. The tradeoff between table size and stretch almost matches the lower bound in Theorem 2. Actually, this trade-off applies to every graph having *linear local tree-width*, a much larger class of graphs including for instance all bounded-genus graphs.

A graph G with n nodes has *local tree-width* $\tau(r)$ if the subgraph induced by nodes within distance r of any node has tree-width at most $\tau(r)$. The local tree-with is *linear* if $\tau(r) = O(r)$. Planar graphs of radius[4] r have tree-width $\leq 3r$, and for graphs of genus γ the tree-width is $O(\gamma r)$ [29], so all these graphs have linear local tree-width. More generally, all *apex-minor-free* graphs[5] have linear local tree-width [21]. These latter graphs can be recognized in linear time [44], and play an important role in Graph Minor Theory with important algorithmic applications [22]. The class of graphs with local tree-width is however not restricted to minor-closed families: bounded degree-d graphs have local tree-with $\tau(r) = O(d^r)$, and d-dimensional meshes have local tree-width $\tau(r) = O(r^d)$.

Theorem 3. *Every connected, unweighted graph of linear local tree-width on n nodes has a labeled routing scheme constructible in polynomial time with $o(\lg^2 n)$-bit addresses, table size $\tilde{O}(n^{1/3})$, and additive stretch $\tilde{O}(n^{1/3})$.*

Our routing scheme is actually more general and it works for any graph G (with worse guarantees on the table size). The additive stretch and the table size bounds rely on a node partition of G and a clustering of it. An (r,σ)-*cell partition* of $G = (V,E)$ is a partition $\{V_i\}$ of its node-set V into σ parts such that each V_i, called *cell*, contains at least $r/2$ nodes and induces a subgraph of radius at most r. By definition, every (r,σ)-cell partition requires $\sigma \leq 2n/r$. A (δ,τ)-*clustering* of a cell partition $\{V_i\}$ is a collection $\{C_i\}$ of connected subgraphs of G, called *clusters*, such that:

1. every node of G belongs to at most δ clusters;
2. the tree-width of any cluster is at most τ; and
3. for every cell V_i there is a cluster C_i containing all shortest paths in G between nodes of V_i.

Thus, every shortest path in the subgraph C_i between two nodes of V_i is a shortest path in G.

The features of our general scheme are summarized as follows:

Theorem 4. *Given an (r,σ)-cell partition with (δ,τ)-clustering of a connected unweighted graph with n nodes, one can construct in polynomial time a labeled routing scheme time with $o(\lg^2 n)$-bit addresses, table size $\tilde{O}(\sigma/r + \delta\tau)$, and additive stretch $O(r\lg\sigma)$.*

Let us first show that Theorem 3 is a direct corollary of Theorem 4.

It is not difficult to see that every connected graph G has an $(r, 2n/r)$-cell partition constructible efficiently (see Lemma 2 in Appendix B). Then, if G has linear local tree-width, we can construct a $(O(\lg n), O(r\lg n))$-clustering based on a *sparse cover* of G, a notion introduced by Awerbuch and Peleg [9], closely related to the (δ,τ)-clustering definition.

A (ρ,d,s)-sparse cover is a collection of connected subgraphs $\{G_i\}$ of G such that:

1. every node of G belongs to at most d subgraphs;
2. the radius of each G_i is at most $s\rho$; and
3. for each node of G at least one G_i contains all neighbors within distance ρ;

[4] A graph has radius r if it has a spanning tree of depth r.

[5] That is the graphs excluding some apices as minor. An apex is a graph with one vertex whose removal leaves a planar graph. K_5 and $K_{3,3}$ are apices, so planar graphs are apex-minor-free.

For general graphs and for any k, ρ there are polynomial-time constructions of $(\rho, O(kn^{1/k}), 2k-1)$-sparse covers. This leads to the construction of $(\rho, O(\lg n), O(\lg n))$-sparse covers by taking $k = \lg n$. Sparse covers can be refined for planar graphs [15], minor-free graphs [5], and graphs of bounded doubling dimension [2]. All these graphs support $(\rho, O(1), O(1))$-sparse covers.

An important observation is that, if G has local tree-width $\tau(r)$, then a $(2r, d, s)$-sparse cover is also a $(d, \tau(2rs))$-clustering (see Lemma 3 in Appendix C). Choosing $d = s = O(\lg n)$, it follows that G has an $(O(\lg n), \tau(O(r \lg n)))$-clustering, which, by linearity of τ, is a $(O(\lg n), O(r \lg n))$-clustering.

In particular, plugging $r = n^{1/3}$ in Theorem 4, wet get $\sigma = O(n^{2/3})$, $\tau = O(n^{1/3} \lg n)$, and $\delta = O(\lg n)$. The additive stretch is $O(r \lg \sigma) = \tilde{O}(n^{1/3})$ and the routing tables have length $\tilde{O}(\sigma/r + \delta\tau) = \tilde{O}(n^{1/3})$, as claimed.

The remainder of this section is dedicated to prove Theorem 4.

3.1 Overview of the Scheme

We start with any (r, σ)-cell partition $\{V_i\}$ and a (δ, τ)-clustering $\{G_i\}$ of G. With each cell V_i, we associate a rooted spanning tree of $G[V_i]$, denoted by T_i, and of depth no more than r. The root of T_i, denoted by c_i, is called the *center* of V_i.

The address of any node u of G is a pair (i, ℓ_u) composed of the unique index i such that $u \in V_i$, and a label ℓ_u allowing shortest-path routing in the tree T_i. According to [31, 55], given the labels ℓ_u, ℓ_v of nodes $u, v \in T_i$, it is possible to compute the next hop on the unique path from u to v in T_i, i.e., the port number leading to a neighbor of u on the path towards v. The length of ℓ_u is $O(\lg^2 |V_i| / \lg\lg |V_i|)$ bits, and thus the length of the address (i, ℓ_u) is $o(\lg^2 n)$ bits.[6]

As one component of our scheme, each node uses a *local* shortest-path routing scheme for all targets in its cluster. A node thus stores the routing information for this local scheme for each cluster C_i it belongs to. Recall that, by definition of the clustering, each node is in at most δ clusters. Tables for these local routing schemes require $o(\tau \lg^2 n)$ bits. Routing between nodes in the same cell is achieved by these local routing schemes as we guarantee that any shortest path between two nodes in the same cell is totally contained in at least one cluster. Note that the shortest paths between nodes of a cell (as opposed to a cluster) may leave and reenter the cell multiple times.

Routing between different cells (say between two cell centers) is achieved by encoding (in the message header) a summary — which we call *trail* — of a "cell path." As paths within cells are handled by the local routing schemes, trails only encode the inter-cell edges of the cell path. A source center has $\sigma - 1$ potential target centers, and each trail may contain $\Omega(\sigma)$ edges. Remembering all these trails potentially requires $\Omega(\sigma^2)$ bits. Since this quantity is too much for one source center, we use a specific collection of trails (simultaneously for all target centers) that can collectively be encoded within $\tilde{O}(\sigma)$ bits. This is done at the price of increasing the additive stretch of the trails.

Tables of each node are restricted to roughly $\tilde{O}(\sigma/r)$ bits, which is r times less than the information required to store all the trails originating at a given source center. As a first step, a particular routing scheme is in charge of *collecting* the routing information, distributed to $\Omega(r)$ nodes within the cell. Since cells have $\geq r/2$ nodes, and radius $O(r)$, the scheme can use a walk of length $O(r)$ to collect all the information.

Then, the trail leading to the target is extracted form this routing information, and the message is sent along the trail. The message is routed between the trail edges using the local routing scheme (inside the clusters).

Summary.

When sending a message from $s \in V_i$ to $t \in V_j$, the following steps are performed (see also Figure 2):

1. from s we route to its center c_i using tree T_i;
2. the routing table of node c_i contains the encoding of a walk of length $O(r)$, which we follow to collect the $\tilde{O}(\sigma)$ bits of routing information on trails (Section 3.3);
3. from the address of t we extract its center c_j, to which we compute the trail from the information collected at c_i (Section 3.2);
4. we route the message along the trail from c_i to c_j by alternatively using edges of the trail and the local routing schemes; and
5. we route to the final destination t using tree T_j.

3.2 Trail Routing

We assume that the routing task is to send messages between centers only, i.e., we skip Step 1 and 5 of the general routing scheme. The routing between arbitrary nodes reduces to this problem, up to additive stretch $O(r)$ using the trees within the cell.

We consider a source center $c_s \in V_s$. Let T be any tree rooted at c_s spanning all the centers of the partition. For a tree T we define two parameters important for the analysis of our scheme: its *distortion*, and its number of *gates*. Tree T has distortion d if for every center c, $d_T(c_s, c) \leq d_G(c_s, c) + d$. A node u in cell V_i is a *gate* if $u = c_i$, or u has neither a proper ancestor nor a proper descendant in V_i. In other words, on the path from c_s to u in T, u is either the first node entering V_i or the last node leaving V_i.

Routing from c_s to any destination center $c_t \in V_t$ can be done using the subpath from c_s to c_t in T, denoted by $T[c_t]$. The *trail* of $T[c_t]$ is the sequence of gates encountered from c_s to c_t.

Consider two consecutive gates of the trail of $T[c_t]$. If u, v belong to the same cluster V_i, then the routing from u to v is performed using the local routing scheme in cluster C_i (see Lemma 4 in Appendix D). It uses shortest paths, and tables of $\tilde{O}(\tau)$ bits per node in C_i, and requires a piece of *advice* of $o(\lg^2 n)$ bits about u, v. This advice is attached to the trail collected at c_s. If u, v belong to different cells, then they must be adjacent, and the port number of this edge is also attached to the trail. Therefore, the information required at c_s only depends on the number of gates in $T[c_t]$.

[6] In this paper, we assume that port numbers of all the edges are fixed (by some adversary) before the labeling of the tree. The label length can be reduced to $(1 + o(1)) \lg_2 |V_i|$ if the designer of the scheme is allowed to permute the port numbers of T_i. In this model, and since all the trees are disjoint, the address length is only $(1 + o(1)) \lg_2 n$ bits by ordering cells according to their number of nodes, so that (i, ℓ_u) has length $\lg_2(i) + (1 + o(1)) \lg_2(n/i)$.

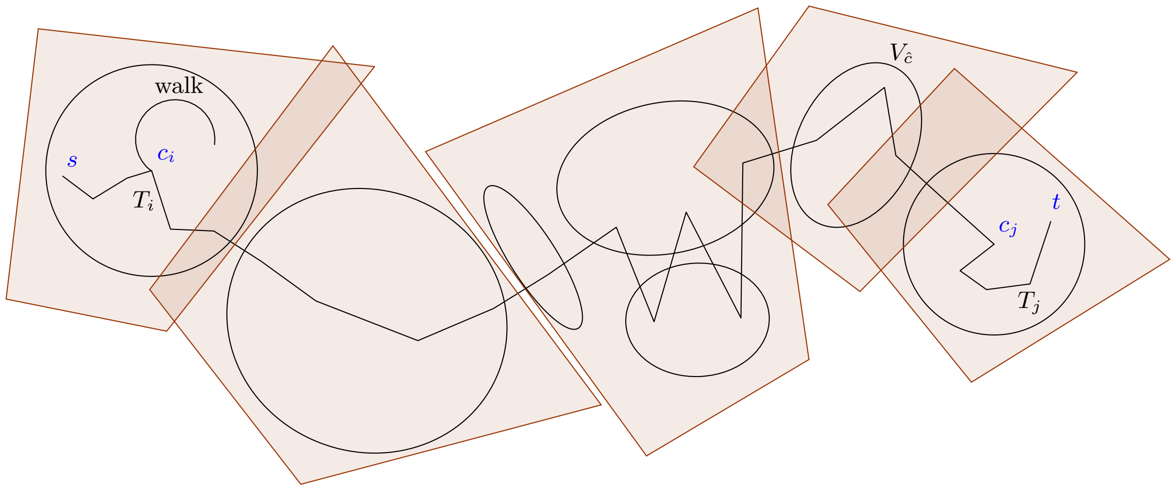

Figure 2: Trail Routing Example: from source s to center c_i on tree T_i, on walk to collect routing information, on *trail* to c_j using the internal scheme within cells and gates between cells, and then on T_j to t.

If T is constructed as a shortest-path tree (without any distortion), it may occur that $T[c_s]$ contains $\Omega(\sigma)$ gates. The number of gates for T can also be as large as $\Theta(\sigma^2)$. This is due to the fact that many paths of T can cross the same cell using different gates. For our purposes, this would be too large. Clearly, there are trees with only $O(\sigma)$ gates, based on spanning trees of the cell graph (nodes correspond to cells, nodes corresponding to neighboring cells are connected by an edge). However, such trees with $O(\sigma)$ gates may have a very large distortion.

The goal of this section is to show that there are trees with both small distortion and only few gates. More precisely, we prove the following.

LEMMA 1. *For every center c, there is a tree rooted at c spanning all the centers with distortion $O(r \lg \sigma)$ and with $O(\sigma \lg \sigma)$ gates.*

PROOF. We fix a center $c \in C$, we run a breadth-first search in G from c and we cut all subtrees that do not contain any center $c' \in C$. Let B denote this tree (spanning C). Although the number of clusters is $|C| = \sigma$, the number of gates may be $\Omega(n)$. For each center $c' \in C$ there is a unique path $T_{c,c'}$ in B from c to c'.

Intuition: to reduce the number of gates, we *merge* some paths at each cluster such that

- each path gets merged with another path at most $\lceil \lg_2 \sigma \rceil$ times, and
- the number of gates from any cluster towards the root is at most $\lceil \lg_2 \sigma \rceil$.

The *merge* operation works as follows: given two paths $T_{c,c'}, T_{c,c''}$ to be merged at a cluster $V_{\hat{c}}$, we keep $T_{c,c'}$ and we replace $T_{c,c''}$ by the concatenation of three paths:

1. the first part of the path $T_{c,c'}$ before it enters $V_{\hat{c}}$ at gate g',
2. the second part of the path $T_{c,c''}$ after it leaves $V_{\hat{c}}$ at gate g', and
3. any path of length $\leq 2r$ in $V_{\hat{c}}$ (not necessarily in B) connecting g' to g''.

We use the best gate g'' to leave $V_{\hat{c}}$ but we may use a sub-optimal gate g' to enter $V_{\hat{c}}$ since the first part of $T_{c,c'}$ is not necessarily optimal for c''. Using the wrong edge to enter $V_{\hat{c}}$ adds a detour of length at most $2r + 2r$, where $2r$ is the diameter of $V_{\hat{c}}$ and another $2r$ is from the potential detour going through g' (using the triangle inequality).

Note that all the gates of $T_{c,c''}$ above $V_{\hat{c}}$ are not needed anymore (as long as they do not appear in other paths) since $T_{c,c''}$ has been merged with $T_{c,c'}$.

We now describe *which paths* to merge. We assign a weight to each path T, corresponding to the number of centers reached through T. At the beginning, all the weights are set to 1. Throughout the merge process, all the weights are powers of two. We start at all the leafs of B (the centers) and merge while proceeding towards the root. Whenever we are at a cluster $V_{\hat{c}}$ where two paths T_1, T_2 with the same weight 2^i meet, we merge them. In this merge, path T_1 is assigned weight $2^i + 2^i = 2^{i+1}$ (and T_2 is not considered anymore). This merging process might continue recursively until all the paths going out of $V_{\hat{c}}$ towards the root have distinct weights. Consequently, at most $\lceil \lg_2 \sigma \rceil$ trails can proceed towards the root, providing a bound on the number of gates towards the root. Since each cell has at most $\lceil \lg_2 \sigma \rceil$ gates towards the root, the total number of gates is at most $O(\sigma \lg \sigma)$.

Since each path is merged at most $\lceil \lg_2 \sigma \rceil$ times (at most once for each i), the length of the path has distortion at most $O(r \lg \sigma)$, as claimed. □

3.3 Collecting the Routing Information

By definition of the (r, σ)-cell partition, each cell contains at least $r/2$ nodes and induces a subgraph of radius at most r. We distribute the routing information evenly among $r/2$ nodes on a walk of length r from the center (using an Euler tour of a tree spanning the $r/2$ closest nodes of the center).

4. CONCLUSION

We prove that routing with additive stretch requires large table sizes — even for graphs as restricted as planar graphs. Routing thus requires a lot of information stored in large tables even for sparse graphs. The existence of dense graphs with large girth was not the only reason for the "hardness" of compact routing. Our lower bounds separate spanner problems from routing problems and give a reason why not more additive routing schemes have been found yet.

Despite these negative results we also provide a new additive compact routing scheme that almost matches the lower bound for planar graphs. Our scheme is also the first additive routing scheme for planar graphs (and, more generally, graphs with linear local tree-width), for which there are compact routing schemes with multiplicative stretch $1+\epsilon$ and table sizes $\widetilde{O}(\epsilon^{-1})$ bits. By our new lower bound we now know that tables of this size imply additive stretch $\tilde{\Omega}(\sqrt{\epsilon n}\,)$.

Although the stretch-space tradeoff of our new scheme almost matches the lower bound, there is also room for improvement: our scheme is rather complicated and the header length is not yet satisfactory (also, headers may require updates on the way). Our lower bound currently does not incorporate headers. The upper bound almost matches the bound on all the parameters involved. Modeling and giving lower bounds on headers remains an open problem.

5. REFERENCES

[1] I. Abraham and C. Gavoille. Object location using path separators. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing, PODC 2006, Denver, CO, USA, July 23-26, 2006*, pages 188–197, 2006.

[2] I. Abraham, C. Gavoille, A. V. Goldberg, and D. Malkhi. Routing in networks with low doubling dimension. In *26th IEEE International Conference on Distributed Computing Systems (ICDCS 2006), 4-7 July 2006, Lisboa, Portugal*, page 75, 2006.

[3] I. Abraham, C. Gavoille, and D. Malkhi. Compact routing for graphs excluding a fixed minor. In *DISC*, pages 442–456, 2005.

[4] I. Abraham, C. Gavoille, D. Malkhi, N. Nisan, and M. Thorup. Compact name-independent routing with minimum stretch. *ACM Transactions on Algorithms*, 4(3), 2008. Announced at SPAA 2004.

[5] I. Abraham, C. Gavoille, D. Malkhi, and U. Wieder. Strong-diameter decompositions of minor free graphs. In *SPAA 2007: Proceedings of the 19th Annual ACM Symposium on Parallel Algorithms and Architectures, San Diego, California, USA, June 9-11, 2007*, pages 16–24, 2007.

[6] I. Althöfer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993.

[7] M. Arias, L. Cowen, K. A. Laing, R. Rajaraman, and O. Taka. Compact routing with name independence. *SIAM J. Discrete Math.*, 20(3):705–726, 2006. Announced at SPAA 2003.

[8] B. Awerbuch, A. B. Noy, N. Linial, and D. Peleg. Improved routing strategies with succinct tables. *Journal of Algorithms*, 11(3):307–341, 1990.

[9] B. Awerbuch and D. Peleg. Sparse partitions (extended abstract). In *31st Annual Symposium on Foundations of Computer Science, 22-24 October 1990, St. Louis, Missouri, USA*, pages 503–513, 1990.

[10] S. Baswana, T. Kavitha, K. Mehlhorn, and S. Pettie. Additive spanners and (α,β)-spanners. *ACM Transactions on Algorithms*, 7(1), Nov. 2010.

[11] F. Bazzaro and C. Gavoille. Localized and compact data-structure for comparability graphs. *Discrete Mathematics*, 309(11):3465–3484, June 2009.

[12] B. Bollobás, D. Coppersmith, and M. L. Elkin. Sparse distance preservers and additive spanners. In *Symposium on Discrete Algorithms (SODA)*, 2003.

[13] A. Brady and L. J. Cowen. Compact routing on power law graphs with additive stretch. In 8^{th} *Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 119–128, Jan. 2006.

[14] A. Brady and L. J. Cowen. Exact distance labelings yield additive-stretch compact routing schemes. In 20^{th} *International Symposium on Distributed Computing (DISC)*, volume 4167 of Lecture Notes in Computer Science, pages 339–354. Springer, Sept. 2006.

[15] C. Busch, R. LaFortune, and S. Tirthapura. Improved sparse covers for graphs excluding a fixed minor. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Principles of Distributed Computing,*

PODC 2007, Portland, Oregon, USA, August 12-15, 2007, pages 61–70, 2007.

[16] W. Chen, C. Sommer, S.-H. Teng, and Y. Wang. Compact routing in power-law graphs. In *Distributed Computing, 23rd International Symposium, DISC 2009, Elche, Spain, September 23-25, 2009. Proceedings*, pages 379–391, 2009.

[17] V. D. Chepoi, F. F. Dragan, B. Estellon, M. Habib, and Y. Vaxès. Additive spanners and distance and routing labeling schemes for hyperbolic graphs, 2009. preprint.

[18] V. D. Chepoi, F. F. Dragan, and Y. Vaxès. Distance and routing labeling schemes for non-positively curved plane graphs. *Journal of Algorithms*, 61(2):60–88, 2006.

[19] E. Cohen. Fast algorithms for constructing t-spanners and paths with stretch t. *SIAM Journal on Computing*, 28(1):210–236, 1998. Announced at FOCS 1993.

[20] L. Cowen. Compact routing with minimum stretch. *Journal of Algorithms*, 38(1):170–183, 2001. Announced at SODA 1999.

[21] E. D. Demaine and M. Hajiaghayi. Equivalence of local treewidth and linear local treewidth and its algorithmic applications. In 14^{th} *Symposium on Discrete Algorithms (SODA)*, pages 840–849. ACM-SIAM, Jan. 2004.

[22] E. D. Demaine, M. Hajiaghayi, and K. Kawarabayashi. Approximation algorithms via structural results for apex-minor-free graphs. In 36^{th} *International Colloquium on Automata, Languages and Programming (ICALP)*, volume 5555 of Lecture Notes in Computer Science, pages 316–327. Springer, July 2009.

[23] Y. Dourisboure. Compact routing schemes for bounded tree-length graphs and for k-chordal graphs. In *Distributed Computing, 18th International Conference, DISC 2004, Amsterdam, The Netherlands, October 4-7, 2004, Proceedings*, pages 365–378, 2004.

[24] Y. Dourisboure and C. Gavoille. Improved compact routing scheme for chordal graphs. In *Distributed Computing, 16th International Conference, DISC 2002, Toulouse, France, October 28-30, 2002 Proceedings*, pages 252–264, 2002.

[25] F. F. Dragan and I. Lomonosov. On compact and efficient routing in certain graph classes. *Discrete Applied Mathematics*, 155(11):1458–1470, 2007.

[26] F. F. Dragan, C. Yan, and D. G. Corneil. Collective tree spanners and routing in AT-free related graphs. In 30^{th} *International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 3353 of Lecture Notes in Computer Science. Springer, June 2004. 68-80.

[27] M. L. Elkin and D. Peleg. $(1+\epsilon,\beta)$-spanner constructions for general graphs. *SIAM Journal on Computing*, 33(3):608–631, 2004.

[28] M. Enachescu, M. Wang, and A. Goel. Reducing maximum stretch in compact routing. In *INFOCOM 2008. 27th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 13-18 April 2008, Phoenix, AZ, USA*, pages 336–340, 2008.

[29] D. Eppstein. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27(3):275–291, 2000.

[30] U. Feige, M. T. Hajiaghayi, and J. R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM J. Comput.*, 38(2):629–657, 2008. Announced at STOC 2005.

[31] P. Fraigniaud and C. Gavoille. Routing in trees. In *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, pages 757–772, 2001.

[32] P. Fraigniaud and C. Gavoille. A space lower bound for routing in trees. In *STACS 2002, 19th Annual Symposium on Theoretical Aspects of Computer Science, Antibes - Juan les Pins, France, March 14-16, 2002, Proceedings*, pages 65–75, 2002.

[33] P. Fraigniaud, E. Lebhar, and L. Viennot. The inframetric model for the internet. In 27^{th} *Annual IEEE Conference on Computer Communications (INFOCOM)*, pages 1085–1093, Apr. 2008.

[34] G. N. Frederickson and R. Janardan. Designing networks with compact routing tables. *Algorithmica*, 3:171–190, 1988.

[35] C. Gavoille. Routing in distributed networks: overview and open problems. *SIGACT News*, 32(1):36–52, 2001.

[36] C. Gavoille and M. Gengler. Space-efficiency of routing schemes of stretch factor three. *Journal of Parallel and Distributed Computing*, 61(5):679–687, 2001.

[37] C. Gavoille and N. Hanusse. Compact routing tables for graphs of bounded genus. In *Automata, Languages and Programming, 26th International Colloquium, ICALP'99, Prague, Czech Republic, July 11-15, 1999, Proceedings*, pages 351–360, 1999.

[38] C. Gavoille, M. Katz, N. A. Katz, C. Paul, and D. Peleg. Approximate distance labeling schemes. In *Algorithms - ESA 2001, 9th Annual European Symposium, Aarhus, Denmark, August 28-31, 2001, Proceedings*, pages 476–487, 2001.

[39] C. Gavoille and D. Peleg. The compactness of interval routing for almost all graphs. *SIAM Journal on Computing*, 31(3):706–721, 2001.

[40] C. Gavoille, D. Peleg, S. Pérennes, and R. Raz. Distance labeling in graphs. *J. Algorithms*, 53(1):85–112, 2004. Announced at SODA 2001.

[41] C. Gavoille and S. Pérennès. Memory requirement for routing in distributed networks. In 15^{th} *Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 125–133. ACM Press, May 1996.

[42] K. Iwama and A. Kawachi. Compact routing with stretch factor of less than three. *IEICE Transactions*, 88-D(1):47–52, 2005. Announced at PODC 2000.

[43] K. Iwama and M. Okita. Compact routing for flat networks. In *Distributed Computing, 17th International Conference, DISC 2003, Sorrento, Italy, October 1-3, 2003, Proceedings*, pages 196–210, 2003.

[44] K. Kawarabayashi. Planarity allowing few error vertices in linear time. In 50^{st} *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 639–648. IEEE Computer Society Press, Oct. 2009.

[45] G. Konjevod, A. W. Richa, and D. Xia. Optimal scale-free compact routing schemes in networks of low

doubling dimension. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 939–948, 2007.

[46] S. Kutten and D. Peleg. Fast distributed construction of small k-dominating sets and applications. *Journal of Algorithms*, 28(1):40–66, 1998. Announced at PODC 1995.

[47] K. A. Laing. Name-independent compact routing in trees. *Inf. Process. Lett.*, 103(2):57–60, 2007. Announced at PODC 2004.

[48] K. A. Laing and R. Rajaraman. A space lower bound for name-independent compact routing in trees. *Journal of Interconnection Networks*, 8(3):229–251, 2007. Announced at SPAA 2005.

[49] H.-I. Lu. Improved compact routing tables for planar networks via orderly spanning trees. *SIAM J. Discrete Math.*, 23(4):2079–2092, 2010. Announced at COCOON 2002.

[50] D. Peleg. Proximity-preserving labeling schemes. *J. Graph Theory*, 33(3):167–176, 2000. Announced at WG 1999.

[51] D. Peleg and A. A. Schäffer. Graph spanners. *Journal of Graph Theory*, 13(1):99–116, 1989.

[52] D. Peleg and E. Upfal. A trade-off between space and efficiency for routing tables. *Journal of the ACM*, 36(3):510–530, July 1989. Announced at STOC 1988.

[53] S. Pettie. Low distortion spanners. In *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings*, pages 78–89, 2007.

[54] M. Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *J. ACM*, 51(6):993–1024, 2004. Announced at FOCS 2001.

[55] M. Thorup and U. Zwick. Compact routing schemes. In *SPAA*, pages 1–10, 2001.

[56] M. Thorup and U. Zwick. Approximate distance oracles. *Journal of the ACM*, 52(1):1–24, 2005. Announced at STOC 2001.

[57] M. Thorup and U. Zwick. Spanners and emulators with sublinear distance errors. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, Miami, Florida, USA, January 22-26, 2006*, pages 802–809, 2006.

[58] D. P. Woodruff. Lower bounds for additive spanners, emulators, and more. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 389–398, 2006.

[59] D. P. Woodruff. Additive spanners in nearly quadratic time. In 37^{th} *International Colloquium on Automata, Languages and Programming (ICALP)*, volume 6198 of Lecture Notes in Computer Science (ARCoSS), pages 463–474. Springer, July 2010.

APPENDIX

A. PLANAR LOWER BOUND

Theorem 2.

For connected, unweighted bounded degree planar graphs on n nodes, any labeled routing scheme using addresses of poly-logarithmic length and routing tables of size μ bits per node has worst-case additive stretch at least $\beta = \tilde{\Omega}(\sqrt{n}/\mu)$.

Proof. Our family of worst-case graphs is a combination of the construction for general graphs (Theorem 1, see also Fig. 1) and techniques by Abraham et al. [2]. For each pair of source s_i and target t_j there is a shortcut in the "skew mesh" *either* on the left-hand-side *or* on the right-hand-side.

"Skew mesh" construction(s).

We closely follow the exposition in [2, Theorem 7]. The final graph will be unweighted. To simplify the exposition, we begin our description with a weighted graph. Let p, q, ρ be positive integers. Let M be a $(p+1) \times (q+1)$ weighted mesh. Each edge in row i has weight $2i$ and each edge in row j has weight $2j$. Due to these weights, the unique shortest path from $(i, q+1)$ to $(p+1, j)$ consists of edges from column i and row j *only*. The shortest path length from $(i, q+1)$ to $(p+1, j)$ is $2i(q+1-j) + 2j(p+1-i)$. Let L be a mesh based on M with some "half-diagonals" (i, j) added as follows. First, each weighted edge is subdivided into two edges of length 1 and $2i - 1$, respectively, where the shorter edge is assigned to the part closer to the origin $(1, 1)$. Second, independent of all other half-diagnolas, the non-identical endpoints of these adjacent edges with weight 1 (the two newly added nodes) can be connected by a new edge with weight 1. If diagonal (i, j) is included, the shortest path length from $(i, q+1)$ to $(p+1, j)$ is reduced by 1. Let R be the mesh built from M containing the half-diagonals that were not added to L. Each half-diagonal (i, j) is thus *either* in L *or* in R. The weighted graphs L, R have $\Theta(pq)$ nodes and $\Theta(pq)$ edges. In their unweighted form (by replacing each edge with integral weight w by w edges), the graphs have $\Theta(pq(p+q))$ nodes and edges. We further subdivide each edge into ρ edges — the resulting graphs have $\Theta(pq(p+q)\rho)$ nodes and edges.

In the final step, we combine these skew meshes with the construction for general graphs (proof of Theorem 1). We add p sources s_i and q targets t_j; we connect each s_i to $l_{i,q+1}$ and $r_{i,q+1}$ using a path with ρ edges and each t_j to $l_{p+1,j}$ and $r_{p+1,j}$ (also using a path with ρ edges[7]). This construction adds another $\Theta((p+q)\rho)$ nodes and edges. Now, for any source s_i and target t_j, if a message is routed using the mesh that does not contain the diagonal (i, j), the route length increases by at least ρ.

In the construction for planar graphs, we have that the number of nodes is $n = \Theta(pq \max\{p, q\}\rho)$. The lower bound changes accordingly. For poly-logarithmic α, we have $p = o(q)$ and thus $n = \Theta(pq^2\rho)$. We obtain (using Eq. (3))

$$\begin{aligned} p(2\rho+1)\mu + (p+q)\alpha &\geq pq \\ \rho\mu &= \tilde{\Omega}(q) \\ \rho &= \tilde{\Omega}(\sqrt{n}/\mu). \end{aligned}$$

[7]In the construction for planar graphs, a single edge would be enough.

The statement follows since the additive stretch is at least ρ. $\square$

B. CELL PARTITION

LEMMA 2. *Every connected graph with $n \geq r/2$ nodes has a $(r, 2n/r)$-cell partition computable in polynomial time.*

PROOF. The number of parts of any (r,σ)-partition is no more $2n/r$. Consider any spanning tree T_0 of the graph and rooted at some node u_0. We greedily construct the cells of the partition as follows. Initially, we set $T := T_0$, and $i := 1$. We iteratively select a node $u \in T$ such that T_u (the subtree of T rooted at u) has depth $\leq r$ and contains $\geq r/2$ nodes. If u is found, we let $V_i := T_u$, we update T by removing T_u, and we repeat for the next cell V_{i+1}.

Clearly, all cells constructed as above satisfy the constraints on the radius and on the number of nodes. So, if T is empty at the end of the loop, then we are done. If T is not empty, we then consider the last cell created, say V_i, and u the last node selected such that $V_i = T_u$, and we update $V_i := T_u \cup T$.

Note that u is well-defined. Indeed, if no node u has been selected in T_0, then every proper subtree has $< r/2$ nodes. It follows that T_0 has depth $\leq r$ (actually depth $\leq r/2$). Since $n \geq r/2$, node u_0 could have been selected in T_0: contradiction.

Cell V_i contains T_u, so it has at least $r/2$ nodes. We need to check that the radius of V_i is $\leq r$. If the depth of T of depth is $\geq r/2$, then T must contains a node w such that T_w has depth $\leq r$ and contains $\geq r/2$ nodes (in particular a node at distance $r/2$ from a leaf of maximum depth in T). So, the depth of T is $< r/2$, and so the distance from u to u_0 is $\leq r/2$. It follows that the distance in T_0 from u to any node $w \in T$ is $\leq r$. Therefore, the radius of V_i is $\leq r$, as claimed. $\square$

C. SPARSE COVERS

LEMMA 3. *If G has local tree-width $\tau(r)$, then any $(2r, d, s)$-sparse cover is also a $(d, \tau(2rs))$-clustering of a (r, σ)-cell partition.*

PROOF. Consider a $(2r, d, s)$-sparse cover $\{G_j\}$ and $(d, \tau(2rs))$-clustering $\{C_i\}$ of G for some (r,σ)-cell partition $\{V_i\}$. It is enough to let for C_i the subgraph G_j covering the $2r$-radius ball around center $c_i \in V_i$.

Clearly, each node belongs to at most d cluster C_i. The radius of G_j is no more than $2rs$, so the tree-with of C_i is bounded by $\tau(2rs)$.

Consider any shortest path P in G between $x, y \in V_i$, and let $u \in P$. Using a path from u to c_i going thru x, we get

$$d_G(u, c_i) \ \leq \ d_P(u, x) + d_{G[V_i]}(x, c_i) \ \leq \ d_G(u, x) + r$$

since P is a shortest path in G and $G[V_i]$ has radius at most r. Similarly, using a path from u to c_i going thru y, we get $d_G(u, c_i) \leq d_G(u, y) + r$. It follows that:

$$d_G(u, c_i) \ \leq \ \min\{d_G(u,x), d_G(u,y)\} + r \ \leq \ \frac{1}{2}(d_G(u,x) + d_G(u,y)) + r \ .$$

We observe that $d_G(u,x) + d_G(u,y) = d_G(x,y) \leq 2r$. If follows that $d_G(u, c_i) \leq 2r$, and thus $u \in C_i$, and so path P is wholly included in C_i as required. $\square$

D. LOCAL ROUTING

LEMMA 4. *Let G be a connected (weighted) graph with n nodes and tree-with τ. There is a routing scheme for G, constructible in polynomial time, with $O(\lg n)$-bit addresses and routing tables of $\tilde{O}(\tau)$ bits such that routing from any source s to any target t can be done along a shortest path provided an advice $A(s,t)$ given at s of $O(\lg^2 n)$ bits.*

This result is based on the classical shortest path routing in tree-width τ graphs. However, the classical solution requires addresses of $\tilde{O}(\tau)$. This is too much, since we need to store target addresses for each gates of our trails. Items of only $\tilde{O}(1)$ are allowed to specify a gate.

PROOF. We first compute a decomposition of G into small pieces using balanced separators (sets of nodes separating the graph into components of size roughly half). If G has tree-width τ, then a decomposition with separators of size $O(\tau\sqrt{\lg \tau})$ can be done in polynomial time [30].

Each node u stores a hierarchy H_u of $O(\lg n)$ separators, and for each node in these separators, it stores the port number leading to it along a shortest path. In total, the routing table for u has length $O(\tau\sqrt{\lg \tau}\lg^2 n) = \tilde{O}(\tau)$ bits. The hierarchy of $O(\lg n)$ separators is chosen such that any two nodes u, v share at least one separator of the hierarchy, i.e., $H_u \cap H_v \neq \emptyset$.

Consider a shortest path P from s to t. Similarly to the cell partition (cf. Section 3), we consider each separator in the set $H_s \cup H_t$ as a cell. And, analogously, we call the first node of P entering and the last one leaving a separator a *gate*. (Note however that separators may not be disjoint.) The number of gates is at most $|H_s \cup H_t| = O(\lg n)$. Because the hierarchy is shared by all the nodes of P, it follows that a trail specifying the gates of P suffices to route, provided that every node in the graph has a routing table for all the nodes of its hierarchy of separators. Each gate can be specified with a $O(\lg n)$ identifier, so an advice $A(s,t)$ of $O(\lg^2 n)$ bits allows shortest-path routing in G from s to t along P. $\square$