Optimal Spanners for Axis-Aligned Rectangles [1]

Tetsuo Asano [a] Mark de Berg [b] Otfried Cheong [b] Hazel Everett [c] Herman Haverkort [d]
Naoki Katoh [e] Alexander Wolff [f]

[a] *JAIST, Japan*
[b] *TU Eindhoven, the Netherlands*
[c] *LORIA, France*
[d] *Utrecht University, the Netherlands*
[e] *Kyoto University, Japan*
[f] *Universität Karlsruhe, Germany*

---

1. Introduction

Geometric networks arise frequently in our everyday life: road networks, telephone networks, and computer networks are all examples of geometric networks that we use daily. They also play a role in disciplines such as VLSI design and motion planning. Almost invariably, the purpose of the network is to provide a connection between the nodes in the network. Often it is desirable that the connection through the network between any pair of nodes be relatively short. From this viewpoint, one would ideally have a direct connection between any pair of nodes. This is usually infeasible due to the costs involved, so one has to compromise between the quality and the cost of the connections.

For two given nodes in a graph, the ratio of their distance in the graph and their 'direct' distance is called the *dilation* or *stretch factor* for that pair of nodes, and the dilation of a graph is the maximum dilation over all pairs of nodes. For geometric networks, this is more precisely defined as follows. Let S be a set of n points (in the plane, say), and let $\mathcal{G}$ be a graph with node set S. Now the dilation for a pair of points p, q is defined as the ratio of the length of the shortest path in $\mathcal{G}$ between p and q, and the length of the segment pq. (The length of a path is the sum of the lengths of its edges.) Again, the dilation of $\mathcal{G}$ is the maximum dilation over all pairs of points in S. A graph with dilation t is called a *t-spanner*. Ideal networks are t-spanners for small t with small cost.

Spanners were introduced by Peleg and Schäffer [6] in the context of distributed computing, and by Chew [1] in the context of computational geometry. They have attracted much attention since—see for instance the survey by Eppstein [2]. The cost of spanners can be measured according to various criteria. For example, it is sometimes defined as the number of edges (here the goal is to find a spanner with $O(n)$ edges), or as the total weight of the edges (here the goal is to find a spanner whose total weight is a constant times the weight of a minimum spanning tree). Additional properties, such as bounding the maximum degree or the diameter, have been considered as well.

We generalize the notion of spanners to geometric networks whose nodes are rectangles rather than points. Let S be a set of n non-intersecting, axis-parallel rectangles and let E be a set of axis-parallel segments connecting pairs of rectangles. For any two points p, q in the union of the rectangles, the dilation is now the ratio of the length of the shortest rectilinear path in the network between p and q and their L_1-distance. Here a path in the network is a path that stays within the union of the rectangles and the connecting segments. The dilation of the network is the maximum dilation over all pairs p, q. Again, our aim is to construct a network whose dilation is small. To illustrate the concept, imagine one is given a number of rect-

---

Email addresses: `t-asano@jaist.ac.jp` (Tetsuo Asano), `m.t.d.berg@tue.nl` (Mark de Berg), `ocheong@win.tue.nl` (Otfried Cheong), `everett@loria.fr` (Hazel Everett), `herman@cs.uu.nl` (Herman Haverkort), `naoki@archi.kyoto-u.ac.jp` (Naoki Katoh), `awolff@ira.uka.de` (Alexander Wolff).

[1] Part of this research was done during the First Utrecht-Carleton Workshop on Computational Geometry. H.H. acknowledges support by the Netherlands' Organization for Scientific Research (NWO).

angular buildings, which have to be connected by footbridges. It is quite frustrating if, to walk to a room opposite ones own room in an adjacent building, one has to walk all the way to the end of a long corridor, then along the footbridge, and then back again along the corridor in the other building. Hence, one would usually place the footbridge in the middle between buildings. Following this analogy, we will call the rectangles in the input *buildings* from now on, and the connecting segments *bridges*. We call the underlying graph of the network the *bridge graph*.

The generalization we study introduces one important additional difficulty in the construction of a spanner: for points one only has to decide *which* edges to choose in the spanner, but for buildings, one also has to decide *where* to place the bridge between a given pair of buildings. It is the latter problem we focus on in this paper: we assume the topology of the network (the bridge graph) is given, and our only task is to place the bridges so as to minimize the dilation.

Formally, our problem can be stated as follows: we are given a set S of axis-parallel disjoint rectangles (buildings) in the plane, a graph $\mathcal{G}$ with node set S, and for each arc e of $\mathcal{G}$ a *bridge region* Λ_e, an axis-aligned rectangle connecting the two buildings. Buildings may degenerate to segments or points. The bridge graph $\mathcal{G}$ must only have arcs between buildings that can be connected by a horizontal or vertical segment, and may not have multiple edges or loops. The bridge regions must be disjoint from each other and the buildings. Our goal is to find a set of horizontal or vertical bridges lying in the bridge regions that has minimum dilation.

Figure 1 shows a bridge graph (the bridge regions are shaded) and a set of possible bridges. Note that the bridge regions Λ_2 and Λ_3 simply allow any bridge between the two buildings, but bridge region Λ_1 has been chosen so as to avoid intersecting s_4 or the bridge between s_3 and s_4.

Our results are as follows.

- In general, the problem is NP-hard.
- If the bridge graph is a tree, then the problem can be solved by a linear program with $O(n^2)$ variables and constraints.
- If the bridge graph is a path, then the problem can be solved in $O(n^3 \log n)$ time.
- If the bridge graph is a path and the buildings are sorted vertically along this path, the problem can be solved in time $O(n^2)$. A $(1+\varepsilon)$-approximation can be computed in linear time.

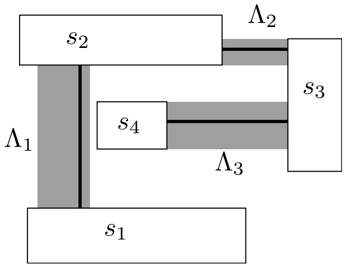

Fig. 1. A bridge graph and a bridge configuration

2. The bridge graph is arbitrary

The bridge-placement problem is NP-hard if the bridge graph is allowed to be arbitrary. We prove this by a reduction from PARTITION. The input to PARTITION is a set B of n positive integers, and the task is to decide whether B can be partitioned into two subsets of equal sum. PARTITION is NP-hard [3, Problem SP12].

Theorem 1 *It is NP-hard to decide whether the bridges in a given bridge graph on n rectangular buildings can be placed such that the dilation is at most* 2.

3. The bridge graph is a tree

In this section we will show that the bridge-placement problem can be solved by a linear program if the bridge graph is a tree. We start by introducing some terminology and notation, and by proving some basic lemmas. As before, we denote the bridge graph by $\mathcal{G}$. Any set of bridges realizing $\mathcal{G}$ will be called a *configuration*.

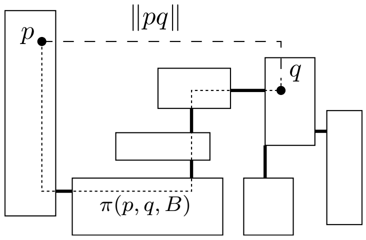

Fig. 2.

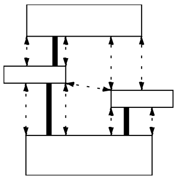

Fig. 3.

Given a configuration B and two points p and q in the union of all buildings, we use $\pi(p,q,B)$ to denote the family of rectilinear shortest paths from p to q within the configuration (that is, paths whose links lie inside buildings or on bridges). The paths of this family are essentially the same, they differ only in how they connect two points inside the same building, and so we will simply speak about *the unique path* $\pi(p,q,B)$. The *dilation* of the path $\pi = \pi(p,q,B)$ is $\mathrm{dil}(\pi) := |\pi|/\|pq\|$, where $|\pi|$ is the total length of π and $\|pq\|$ is the L_1-distance of p and q. Figure 2 shows a configuration and an example path.

The *dilation* $\mathrm{dil}(B)$ *of a configuration* B is defined as the maximum dilation of any path with respect to B. Our aim is to find a configuration of minimum dilation. We first characterize pairs of points that are responsible for the dilation of a given configuration.

Lemma 2 *Let σ be the dilation of a configuration B whose underlying graph is a tree. Then there are points p and q with $\mathrm{dil}(\pi(p,q,B)) = \sigma$ such that the closed bounding box of p and q does not contain any point of a building other than p and q, and at least one of the points p and q is a building corner.*

A point pair (p,q) as in the lemma—its bounding box contains no other point of any building and at least one of p and q is a building corner—will be called a *visible pair*—see Figure 3 for examples. We denote the set of all visible pairs by $\mathcal{V}$.

Given a bridge graph $\mathcal{G}$, our goal is to minimize

$$\max_{(p,q)\in\mathcal{V}} \mathrm{dil}(\pi(p,q,B))$$

over all configurations B realizing $\mathcal{G}$. We show that this problem can be reformulated as a linear program.

Theorem 3 *If the bridge graph $\mathcal{G}$ is a tree, then a placement of the bridges that minimizes the dilation can be computed by solving a linear program with $O(n^2)$ variables and constraints, where n is the number of bridges in the bridge graph.*

4. The bridge graph is a path

In the previous section we have given a linear program for the bridge-placement problem for the case where the bridge graph is a tree. Linear programs can be solved in practice, and for integer coefficients, interior-point methods can solve them in time polynomial in the bit-complexity of the input [4]. It is not known, however, if they can be solved in polynomial time on the real RAM, the standard model of computational geometry. In this section, we give polynomial time algorithms for the case where the bridge graph is a path.

Since the bridge graph $\mathcal{G}$ is a path, we can number the buildings and bridges so that bridge b_i connects buildings s_{i-1} and s_i, for $1 \le i \le n$ (so there are $n+1$ buildings and n bridges). Before we continue, we need to introduce some more terminology. We consider a path $\pi = \pi(p,q,B)$ to be oriented from p to q. After traversing a bridge b, the path can continue straight on to traverse the next bridge b' if b and b' are collinear. In all other cases, it has to turn.

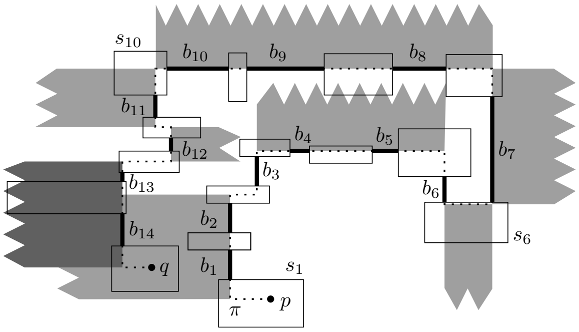

Fig. 4. U-turns and their outer sides

Given a path π, a *link* ℓ of π is a maximal straight segment of the path. A link can contain more than one bridge if they are collinear. For example, in Figure 4 there is a link containing b_1 and b_2, and another link containing b_8, b_9, and b_{10}.

The path π turns at both ends of a link (except for the first and last link). The link is a *right U-turn* if π turns right before and after the link. A *left U-turn* is defined symmetrically. In Figure 4, the links containing bridges (b_1,b_2), (b_4,b_5), and b_{12} are right U-turns, while the links containing b_7, (b_8,b_9,b_{10}), b_{11}, and (b_{13},b_{14}) are left U-turns. Note that there can be U-turns that do not contain any bridges, as the link of π inside building s_6 in Figure 4.

The *inner side* and *outer side* of a U-turn are rectangular regions infinite on one side, and bounded by the line supporting the link and the two lines orthogonal to it through the first and last points of the link. The outer side lies locally to the left of a right U-turn, or to the right of a left U-turn, the inner side lies locally to the right of a right U-turn or to the left of a left U-turn. In Figure 4, the outer sides of all U-turns are shaded.

U-turns are the links of a path that determine its dilation, as the following lemma shows.

Lemma 4 *Let B and B' be configurations, (p, q) a visible pair, and $\pi := \pi(p, q, B)$ and $\pi' := \pi(p, q, B')$ the paths between p and q with respect to the two configurations. If $\mathrm{dil}(\pi') < \mathrm{dil}(\pi)$ then there exists a U-turn ℓ containing $b_i \dots b_j$ of π such that the corresponding bridges $b'_i, \dots, b'_j$ of B' lie strictly on the inner side of ℓ.*

We will give an algorithm that takes as input the set of buildings $s_0, \dots, s_n$ and a real number $\sigma > 1$, and computes a configuration B with $\mathrm{dil}(B) \leq \sigma$, or determines that no such configuration exists.

The algorithm computes n sets $I_1, I_2, \dots, I_n$, where I_i is a set of possible bridges between s_{i-1} and s_i. The sets are defined recursively as follows. Assume that $I_1, \dots, I_{i-1}$ have already been defined. For each visible pair (p, q) with $p \in \bigcup_{j=0}^{i-1} s_j$ and $q \in s_i$ we define $I(p, q)$ as the set of bridges b_i connecting s_{i-1} and s_i such that the following holds: there is a set of bridges $b_1 \in I_1$, $b_2 \in I_2$, $\dots, b_{i-1} \in I_{i-1}$ such that $\mathrm{dil}(\pi(p, q, (b_1, \dots, b_i))) \leq \sigma$. Finally, I_i is the intersection of all $I(p, q)$.

Note that for each visible pair (p, q) we can choose the bridges in $I_1, \dots, I_{i-1}$ independently. This makes it possible to compute I_i efficiently, as we will see below. On the other hand, it implies that not every sequence of bridges chosen from the sets will be a configuration with dilation at most σ—our main lemma will be to show that such a sequence does indeed exist.

Once we know $I_1, \dots, I_n$, we can recursively compute a configuration with dilation at most σ: Choose an arbitrary bridge $b_n \in I_n$. If bridges $b_{n-1}, b_{n-2}, \dots, b_{i+1}$ have been computed, choose a bridge $b_i \in I_i$ whose distance from b_{i+1} is minimal. Since I_i is an "interval of bridges", this implies that either b_i and b_{i+1} are collinear, or b_i is one of the extreme bridges in I_i. We now prove that this approach is correct.

Lemma 5 *Let $I_1, \dots, I_n$ be given as defined above. A configuration B with dilation $\mathrm{dil}(B) \leq \sigma$ exists if and only if $I_n \neq \emptyset$. If it exists, it can be computed in $O(n)$ time from the intervals.*

Lemma 6 *The intervals $I_1, \dots, I_n$ defined above can be computed in $O(n^2)$ time and $O(n)$ space.*

Lemmas 6 and 5 imply the following theorem.

Theorem 7 *Given a bridge graph $\mathcal{G}$ on a set of $n+1$ buildings that is a path and a real number $\sigma > 1$, we can in time $O(n^2)$ compute a configuration B realizing $\mathcal{G}$ with $\mathrm{dil}(B) \leq \sigma$ or determine that no such configuration exists.*

It seems hard to improve this result when there are $\Theta(n^2)$ visible pairs that could determine the dilation. In fact, we do not even know how to decide in $o(n^2)$ time whether a *given* configuration has dilation $\leq \sigma$. If the number k of visible pairs of the given set of buildings is $o(n^2/\log n)$, the running time can be improved to $O(k \log n)$.

We solve the original optimization problem using Megiddo's parametric search [5].

Theorem 8 *Given a bridge graph on a set of $n+1$ buildings that is a path, we can compute a configuration with the optimal dilation in time $O(n^3 \log n)$, or in time $O(nk \log^2 n)$, where k is the number of visible pairs.*

References

[1] L. P. Chew. There are planar graphs almost as good as the complete graph. *J. Comput. Syst. Sci.*, 39:205–219, 1989.

[2] David Eppstein. Spanning trees and spanners. In Jörg-Rüdiger Sack and Jorge Urrutia, editors, *Handbook of Computational Geometry*, pages 425–461. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.

[3] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York, NY, 1979.

[4] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[5] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *J. ACM*, 30(4):852–865, 1983.

[6] D. Peleg and A. Schäffer. Graph spanners. *J. Graph Theory*, 13:99–116, 1989.