On the Spanning and Routing Ratios of the Yao-Four Graph

Prosenjit Bose
Carleton University, Ottawa, Canada

Darryl Hill
Carleton University, Ottawa, Canada

Michiel Smid
Carleton University, Ottawa, Canada

Tyler Tuttle
Carleton University, Ottawa, Canada

Abstract

The Yao graph is a geometric spanner that was independently introduced by Yao [SIAM J. Comput., 1982] and Flinchbaugh and Jones [SIAM J. Algebr. Discret. Appl., 1981]. We prove that for any two vertices of the undirected version of the Yao graph with four cones, there is a path between them with length at most $13 + 5/\sqrt{2} \approx 16.54$ times the Euclidean distance between the vertices, improving the previous best bound of approximately 54.62. We also present an online routing algorithm for the directed Yao graph with four cones that constructs a path between any two vertices with length at most $17 + 9/\sqrt{2} \approx 23.36$ times the Euclidean distance between the vertices. This is the first routing algorithm for a directed Yao graph with fewer than six cones. The algorithm uses knowledge of the coordinates of the current vertex, the (up to) four neighbours of the current vertex, and the destination vertex to make a routing decision. It also uses one additional bit of memory. We show how to dispense with this single bit at the cost of increasing the length of the path to $\sqrt{331 + 154\sqrt{2}} \approx 23.43$ times the Euclidean distance between the vertices.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Yao graph, online routing, geometric spanners

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2024.15

Acknowledgements We would like to thank an anonymous reviewer for suggestions to improve the presentation and for pointing out an error in our original proof of Lemma 7.

1 Background

Online routing is the problem of constructing a path in a graph from some current vertex to a given destination vertex, without knowing the entire graph ahead of time. The path must be constructed one vertex at a time. A *routing algorithm* computes, given some vertex, the next vertex on the path.

The information available to a routing algorithm is of great importance. We say that a routing algorithm is *local* if the only information available to it is knowledge of the current vertex, the immediate neighbours of the current vertex, the destination vertex, plus a constant amount of extra information.

In this paper we will consider routing algorithms for a specific type of geometric graph called a Yao graph. A geometric graph is a graph whose vertex set is a set of points in the plane, and whose edges are weighted by the Euclidean distance between their endpoints. We will consider both directed and undirected graphs. Since the vertices of a geometric graph are points, we will assume that the routing algorithm has access to their coordinates.

Let u and v be two points in the plane. Define $d_x(u,v)$ and $d_y(u,v)$ to be the horizontal and vertical distances between u and v. In this paper we will make use of three different distance functions, or metrics, on $\mathbf{R}^2$. They are the L_1 metric, the L_2 (or Euclidean) metric, and the L_∞ metric.

$$\begin{aligned}\|uv\|_1 &= d_x(u,v) + d_y(u,v)\\ \|uv\|_2 &= \sqrt{d_x(u,v)^2 + d_y(u,v)^2}\\ \|uv\|_\infty &= \max\{d_x(u,v), d_y(u,v)\}\end{aligned} \tag{1}$$

All of the results will be in terms of the standard Euclidean distance, but the analysis will make use of the other metrics.

Let $G = (V, E)$ be a geometric graph, directed or undirected. Let $d_G(u,v)$ denote the length of the shortest path from u to v in G. For a real number t, we say that G is a t-spanner if $d_G(u,v) \le t\|uv\|_2$ for all u and v in V. The *spanning ratio* (also called the stretch factor) of G is the minimum such t.

Let $\mathcal{A}$ be a routing algorithm, and let $d_G^{\mathcal{A}}(u,v)$ denote the length of the path from u to v in G constructed by algorithm $\mathcal{A}$. We say that $\mathcal{A}$ is t-competitive if $d_G^{\mathcal{A}}(u,v) \le t\|uv\|_2$ for all u and v in G. The *routing ratio* of $\mathcal{A}$ is the smallest such t. If $\mathcal{A}$ is t-competitive for all graphs in some class $\mathcal{G}$, then we say that $\mathcal{A}$ is t-competitive on $\mathcal{G}$. The precise definition of an online routing algorithm will be given in Section 1.2.

1.1 Yao graphs

The results of this paper are about the spanning and routing ratios of a specific class of geometric graph known as Yao graphs. We begin by defining these graphs.

Fix an integer $k \ge 3$. Let R_i be the ray emanating from the origin making an angle of $2\pi i/k$ with the positive x-axis[1]. The region between two consecutive rays is called a cone. Specifically, let C_i be the region between R_i and R_{i+1}, including R_i but not R_{i+1}. The k cones C_0 through C_{k-1} partition the plane (minus the origin) into k regions. Let $C_i(p)$ and $R_i(p)$ be C_i and R_i translated so that the rays emanate from p rather than the origin.

Given a set P of n points in the plane, we construct a directed graph in the following way. For each point p in P, we add at most k edges to the graph, one edge per cone $C_i(p)$ of p. Let q be the point in $P \cap C_i(p)$ that minimizes the distance $\|pq\|_2$. Add a directed edge from p to q. See Figure 1 for an example. Repeating this for every point and every cone adds at most kn edges to the graph. The resulting graph is called the (directed) Yao-k graph of P, denoted $\vec{Y}_k(P)$ or simply $\vec{Y}_k$ if the set P is clear from context. The undirected Yao-k graph $Y_k(P)$ (or Y_k if the set P is clear) is obtained by forgetting the direction of each edge of $\vec{Y}_k(P)$.

Notice that the vertices of a directed Yao-k graph have outdegree bounded by k. The vertices of an undirected Yao-k graph can have unbounded degree.

The Yao graph was first defined in the early 1980s independently by Flinchbaugh and Jones [1] and Yao [13]. Althöfer et al. first proved that Yao graphs are spanners in 1993 [6]. Specifically, they showed that for any set P of points and any $t \ge 1$, there is an integer k such that $Y_k(P)$ is a t-spanner of P.

Later work found specific bounds for spanning ratios of Yao graphs. Bose et al. showed that the spanning ratio of $Y_k(P)$ is at most $1/(\cos\theta - \sin\theta)$ for all $k \ge 9$ [11], where $\theta = 2\pi/k$. This was later improved to $1/(1 - 2\sin(\theta/2))$ for all $k \ge 7$ [12]. Finally, Barba et al. showed that the spanning ratio is at most $1/(1 - 2\sin(3\theta/8))$ for odd values of $k \ge 5$ [2].

[1] Angles are measured counterclockwise.

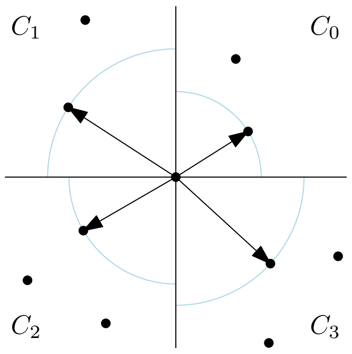

Figure 1 The four cones used to define the $\vec{Y}_4$ graph.

The techniques used to bound the spanning ratio of Yao graphs with many cones do not directly translate to Yao graphs with very few cones. For these, more specific arguments are required. For Y_6, the first published bound was by Damian and Raudonis, who showed that the spanning ratio is at most 17.64 [9]. This was later improved by Barba et al. to 5.8 [2].

In the same paper that gives a bound of $1/(1-2\sin(3\theta/8))$ on the spanning ratio of Y_5, Barba et al. [2] give a more precise argument that the spanning ratio of Y_5 is at most $2+\sqrt{3}$. For Y_4, the first bound was by Bose et al. [12], who showed that the spanning ratio is at most 662. This was improved by Damian and Nelavalli to 54.62 [10].

For fewer than four cones Yao graphs are not necessarily spanners [5]. In fact, the Y_1 graph is equivalent to the (directed) nearest-neighbour graph, which may not even be connected. The Y_2 and Y_3 graphs are connected, but the directed versions are not necessarily strongly connected. These results are summarized in Table 1.

For more than six cones, the proofs that Yao graphs are spanners give a routing algorithm called cone routing: always move to the neighbour in the cone that contains the destination. This gives a routing ratio equal to the spanning ratio. For $\vec{Y}_6$, a local routing algorithm is known with a routing ratio of $1+38/\sqrt{3} \approx 22.94$ [7]. For four or five cones, however, no local routing algorithms are known.

Theta graphs [4, 8] are a class of graphs that are similar to Yao graphs. The construction begins the same way, by partitioning space into cones, but instead of adding an edge from a vertex to its nearest neighbour in each cone, the vertices in each cone are projected onto the bisector of the cone and an edge is added to whichever vertex has the closest projection. Like Yao graphs, Theta graphs are spanners, and routing algorithms for Theta graphs have been studied. In this paper we will present an online routing algorithm for $\vec{Y}_4(P)$ that is a modification of the best-known routing algorithm for $\vec{\Theta}_4(P)$ [3]. Although the algorithm is similar, some care must be taken since every edge of a Theta graph defines an empty triangle, whereas every edge of a Yao graph defines an empty sector of a circle. More importantly, our analysis of the algorithm is novel and we believe it is much simpler than the analysis for the $\vec{\Theta}_4(P)$ routing algorithm.

1.2 Local routing

Let $G=(V,E)$ be a graph. An online routing algorithm on G can be modelled as a function $f : V \times V \times P(V) \times \{0,1\}^* \to V \times \{0,1\}^*$, where $P(V)$ is the power set of V.

The first two parameters of f are the current vertex u and the target vertex v. The third parameter is the set of vertices we can use to make a routing decision. We will focus on *local* routing, where this parameter is the set of neighbours of u. The fourth parameter is a finite bitstring called the *header*, which a routing algorithm can use to store arbitrary information as it constructs a path.

Table 1 Lower and upper bounds for spanning ratios of Yao graphs ($\theta = 2\pi/k$). Our improved upper bound for $k = 4$ is highlighted in bold. In the bottom four rows, m is a positive integer.

Number of cones k	Lower bound	Upper bound
3 or fewer	∞	∞
4	Open	**16.54**
5	2.87	3.74
6	2	5.8
$4m+2$	$1+2\sin(\theta/2)$	$\dfrac{1}{1-2\sin(\theta/2)}$
$4m+3$	$1+2\sin\left(\frac{3\theta}{8}\right)\left(1+2\left(\sin\left(\frac{13\theta}{16}\right)+\sin\left(\frac{19\theta}{16}\right)\right)\frac{\sin(\theta/16)}{\sin(2\theta)}\right)$	$\dfrac{1}{1-2\sin(3\theta/8)}$
$4m+4$	$1+2\sin(\theta/2)(1+\tan(\theta/2))$	$\dfrac{1}{1-2\sin(\theta/2)}$
$4m+5$	$1+2\sin(3\theta/8)+4\sin(5\theta/16)\sin(3\theta/8)$	$\dfrac{1}{1-2\sin(3\theta/8)}$

A routing algorithm must take these four parameters and decide which neighbour of u should come next on the path, and potentially modify the header in some way. These are the two outputs of the function f. A path from a vertex u to a vertex v is constructed in the following way. Let $u_0 = u$ and h_0 be some initial header that the routing algorithm is allowed to compute before constructing the path. Then we get a sequence of vertices and headers defined by $(u_{i+1}, h_{i+1}) = f(u_i, t, N(u_i), h_i)$. If $u_i = v$ for some i, we stop as the routing algorithm has successfully found a path from u to v.

If, for any two vertices u and v in G, a routing algorithm $\mathcal{A}$ constructs a path from u to v, then we say that $\mathcal{A}$ guarantees delivery on the graph G. Let $d_G^{\mathcal{A}}(u,v)$ denote the Euclidean length of the path from u to v in G constructed by algorithm $\mathcal{A}$. We say that $\mathcal{A}$ is t-competitive if $d_G^{\mathcal{A}}(u,v) \le t\|uv\|_2$. The *routing ratio* of $\mathcal{A}$ on G is the smallest t such that $\mathcal{A}$ is t-competitive.

If h_i is the empty bitstring for all i, then we say that $\mathcal{A}$ is *memoryless.* If $h_i = h_0$ for all i, then we say that $\mathcal{A}$ uses a static header. That is, the header is computed before the routing algorithm begins and never modified. Otherwise we say that $\mathcal{A}$ uses a dynamic header. The *memory usage* of a routing algorithm is the maximum length of h_i at any point during construction of a path from u to v, over all pairs of vertices in G.

2 The greedy-sweep algorithm

Let P be a finite set of points in the plane. To avoid tedious case analysis, we will make a general position assumption that no two points have the same x or y coordinate, and that no two points lie on a line with slope $+1$ or -1. Let s and t be two points of P. In this section we will describe the routing algorithm for constructing a path from s to t in the directed graph $\vec{Y}_4(P)$. First, we give some definitions that will be needed to describe and analyze the algorithm.

Let p be any point in the plane. For another point q, define W_{pq} to be the quarter-circle in $C_i(p)$ that is centred at p with q on its boundary contained in $C_i(p)$. Define the *diagonals* of p to be the lines through p with slope $+1$ and -1. Denote the diagonals of p by $d_+(p)$ and $d_-(p)$. The first step is to choose one of the two diagonals of t. Given the positions of s and t, choose whichever diagonal is closer to s. This can be determined by checking which cone of t contains s. If s is in $C_0(t)$ or in $C_2(t)$, then choose $d_+(t)$. Otherwise, choose $d_-(t)$. Let d denote the chosen diagonal. Knowledge of this diagonal will be needed to make routing decisions. The choice of diagonal needs exactly one bit of memory to be remembered. Later we will see that this bit can be dispensed with, at the cost of a slightly higher routing ratio. Finally, define the *height* of a point p, denoted $h(p)$, to be the L_1 distance from p to d.

For any point p of P, exactly one of the cones of p has a nonempty, bounded intersection with d. We say that this cone of p *faces* d. The intersection of the halfplane of d that contains p and the cone forms a right triangle. Denote this triangle by $T(p, d)$.

Algorithm 1 Pseudocode for the greedy-sweep algorithm.

```
                ▷ The procedure GREEDY($p$, $t$, $d$) returns the neighbour of $p$ in the cone that
 contains $t$, and the procedure SWEEP($p$, $t$, $d$) returns the neighbour of $p$ in the cone that
 faces diagonal $d$, or null if such a neighbour does not exist.
 procedure GREEDYSWEEP($p$, $t$, $d$)
     $q \leftarrow$ GREEDY$(p, t, d)$
     $r \leftarrow$ SWEEP$(p, t, d)$
     if $r$ = null or $\|pr\|_2 > h(p)$ then
         return $q$
     else
         return $r$
     end if
 end procedure
```

To make a routing decision at a point p, we have a choice of up to four neighbours. We will determine where to go in the following way. Consider the triangle $T(p, d)$. If $T(p, d)$ is empty of points of P, then we say that it is *clean*, or that p is clean with respect to d. Given the information at p, it might not be possible to determine if $T(p, d)$ is clean or not. However, if r is the neighbour of p in the cone containing $T(p, d)$, and $\|pr\|_2 > h(p)$, then $T(p, d)$ *must* be clean. Also, if p does not have a neighbour in the cone that faces d, then $T(p, d)$ is clean. If either of these two conditions are met, then move from p to the neighbour of p in whichever cone contains t. This is called a *greedy step*. Otherwise, move from p to its neighbour in the cone that faces d. This is called a *sweeping step*. See Figure 2.

In two cones of t, a sweeping step and a greedy step are identical, in the sense that they both select the same neighbour of p. For example, if $d = d_-(t)$ and if p is in $C_0(t)$, then a greedy step and a sweeping step will both choose the neighbour of p in $C_2(p)$. In this case we will consider the move to be a sweeping step. This means that greedy steps are only possible in two cones of t. Which two cones will depend on whether $d = d_-(t)$ or $d = d_+(t)$.

All of the information necessary to make a routing decision can be determined solely from the coordinates of p, the neighbours of p, and t, as well as the slope of the diagonal d. See Algorithm 1 for a pseudocode description of the algorithm.

3 Analysis

We begin by stating a lemma that will be very useful in our analysis, relating the three different metrics under consideration.

▶ **Lemma 1.** *For any points u and v in the plane, the following chain of inequalities holds:*

$$\|uv\|_\infty \le \|uv\|_2 \le \|uv\|_1 \le \sqrt{2}\|uv\|_2 \le 2\|uv\|_\infty. \tag{2}$$

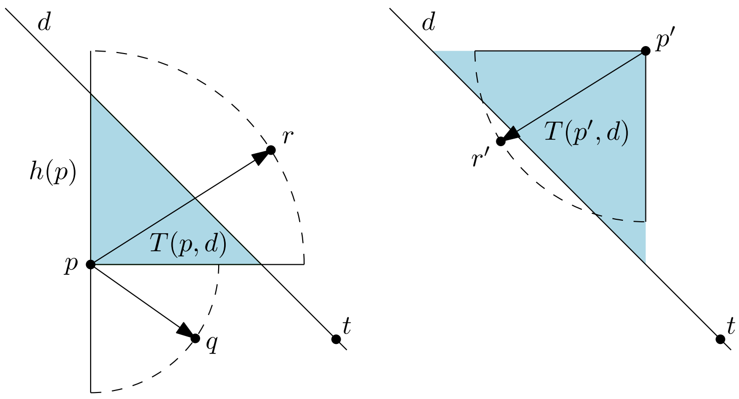

Figure 2 From p, a greedy step to q would be taken since $\|pr\|_2 > h(p)$. This implies that $T(p,d)$ is empty. From p' a sweeping step to r' would be taken since $\|p'r'\|_2 < h(p')$. There is not enough information at p' to determine if $T(p',d)$ is empty or not, since part of $T(p',d)$ is outside of the quarter circle defined by p' and r'.

Let P be a finite set of points in the plane, and let s and t be points in P. In this section we will analyze the path in $\vec{Y}_4(P)$ constructed by the greedy-sweep algorithm starting from s, with t as the destination. We begin the analysis by proving that the greedy-sweep algorithm guarantees delivery, by eventually reaching t. For the remainder of this section, we will assume without loss of generality that s is in $C_1(t)$, so $d = d_-(t)$, and that s is below d.

▶ **Lemma 2.** *Let s and t be different vertices of a $\vec{Y}_4$ graph, and consider the path constructed by the greedy-sweep algorithm starting at s with t as the destination. Let u and v be two consecutive vertices on this path. Then v is inside the square centred at t with u on its boundary, and so $\|ut\|_\infty > \|vt\|_\infty$.*

This lemma immediately implies that the algorithm terminates.

▶ **Corollary 3.** *The greedy-sweep algorithm guarantees delivery on the $\vec{Y}_4$ graph.*

To analyze the routing ratio of the greedy-sweep algorithm, we will consider greedy steps and sweeping steps separately. Recall that d is the diagonal fixed by the algorithm.

Consider an edge uv traversed while routing. Since no two points lie on a common diagonal by our general position assumption, the height change $h(v) - h(u)$ must be either positive or negative. Let D be the set of edges uv such that the height decreases from u to v, meaning $h(u) > h(v)$, and let I be the set of edges such that the height increases from u to v. Since $h(t) = 0$ because t lies on d, we know that the total decrease in height must equal the total increase in height plus the height of the initial point s. That is,

$$\sum_{uv\in D} \big(h(u) - h(v)\big) = h(s) + \sum_{uv\in I} \big(h(v) - h(u)\big). \tag{3}$$

The next two lemmas show that a sweeping step will always result in a height decrease that is at least proportional to the length of the edge, and that the length of a greedy edge is at least equal to the change in height.

▶ **Lemma 4.** *Let uv be an edge taken during a sweeping step. We have $h(u) - h(v) \geq (2 - \sqrt{2})\|uv\|_2$.*

Proof. We will consider two cases. See Figure 3. First, if uv does not cross d, then $h(v) = h(u) - \|uv\|_1$ and we have $h(u) - h(v) = \|uv\|_1 \geq \|uv\|_2$ by Lemma 1.

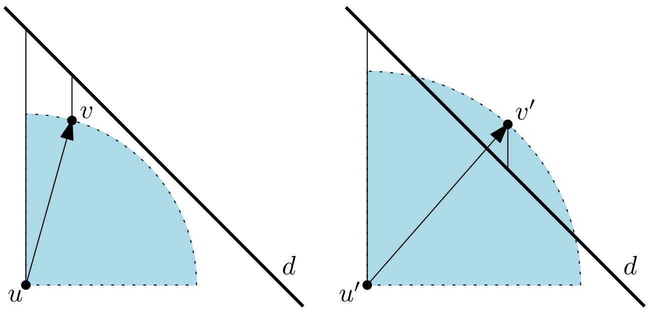

Figure 3 Illustration of Lemma 4. The edge uv does not cross d, and the edge $u'v'$ does. In both cases the height decreases when traversing the edge.

Suppose uv crosses d, and that v is in $C_i(u)$. Let θ be the angle made by the edge uv with the ray $R_i(u)$. We have $h(v) = \|uv\|_1 - h(u)$, so Suppose uv crosses d, and that uv is an i-edge. Let θ be the angle made by the edge uv with the ray $R_i(u)$. We have $h(v) = \|uv\|_1 - h(u)$, so

$$\begin{aligned}h(u) - h(v) &= 2h(u) - \|uv\|_1 \\ &\geq 2\|uv\|_2 - (\sin\theta + \cos\theta)\|uv\|_2 \\ &= (2 - (\sin\theta + \cos\theta))\|uv\|_2 \\ &\geq (2 - \sqrt{2})\|uv\|_2.\end{aligned}$$ ◀

▶ **Lemma 5.** *Let uv be an edge taken during a greedy step. We have $h(v) - h(u) \leq \|uv\|_2$.*

Proof. Assume without loss of generality that u is in $C_1(t)$ and that $d = d_-(t)$. If $h(v) < h(u)$, then the inequality is trivially satisfied. If $h(v) > h(u)$, then we have $h(v) - h(u) = d_y(u, v) \leq \|uv\|_2$. ◀

Now let S be the set of sweeping edges and G be the set of greedy edges. Lemma 4 implies $S \subseteq D$, so we must have $I \subseteq G$. In other words a sweeping edge will always decrease the height, so if an edge increases the height it must be a greedy edge.

▶ **Lemma 6.** *The total length of all the sweeping edges is at most $(1+\frac{1}{\sqrt{2}})(h(s)+\sum_{uv\in G}\|uv\|_2)$.*

Proof. The proof follows from Lemmas 4 and 5, and some simple manipulations:

$$\begin{aligned}\sum_{uv\in S}\|uv\|_2 &\leq \sum_{uv\in D}\frac{1}{2-\sqrt{2}}\big(h(u) - h(v)\big) \\ &= \frac{1}{2-\sqrt{2}}\sum_{uv\in D}\big(h(u) - h(v)\big) \\ &= \frac{1}{2-\sqrt{2}}\left(h(s) + \sum_{uv\in I}\big(h(v) - h(u)\big)\right) \\ &\leq \frac{1}{2-\sqrt{2}}\left(h(s) + \sum_{uv\in G}\|uv\|_2\right).\end{aligned}$$ ◀

Finally we will bound the total length of the greedy edges in terms of $\|st\|_\infty$. For any point u, let $\overline{u}$ be the projection of u onto d. Define the *footprint* of an edge uv in the following way. Let w be the point along either ray bounding the cone $C_i(u)$ that contains v such that $\|uw\|_2 = \|uv\|_2$. The footprint of uv is the segment $\overline{uw}$. See Figure 4. It does not matter which ray we chose since the projection of w would be the same in both cases. Note that we have $\|uv\|_2 = \sqrt{2}\|\overline{uw}\|_2$.

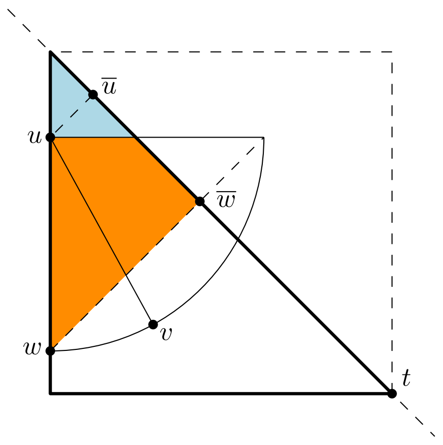

Figure 4 A greedy edge uv. Its footprint is the segment $\overline{uw}$. The blue triangle is empty because uv is a greedy edge, and the orange region is empty because it is contained in W_{uv}, the empty quarter circle defined by the edge uv. For any greedy edge $u'v'$ such that u' is in $C_1(t)$, u' is below d, and u' comes after u in the routing path, we must have u' in the indicated triangle. The footprint of $u'v'$ must be disjoint from the footprint of uv.

Recall that there are only two cones of t where greedy edges can originate, since in two of the cones of t a greedy step would be the same thing as a sweeping step. In both of these cones, the greedy edge can begin either above or below d. Split the set greedy edges into four subsets, depending on which cone of t the edge originates in and whether the edge begins above or below d. We will show that, for any two edges that are in the same subset of G, their footprints are disjoint (except possibly at a single point).

▶ **Lemma 7.** *Let uv and $u'v'$ be two greedy edges such that u and u' are in the same cone of t, and both are either above or below d. Then the footprints of these two edges are disjoint.*

Proof. Let $\overline{uw}$ and $\overline{u'w'}$ be the footprints of the edges uv and $u'v'$, respectively. Assume without loss of generality that u appears before u' on the routing path, and that u and u' are in $C_1(t)$ and below d. We will show that the points $\overline{u}$, $\overline{w}$, $\overline{u'}$, $\overline{w'}$, and t appear in exactly that order along d. To do this, we show that $\overline{uw}$ and $\overline{u'w'}$ are contained in $C_1(t)$ and that

$$\|\overline{u}t\|_1 > \|\overline{w}t\|_1 \geq \|\overline{u'}t\|_1 > \|\overline{w'}t\|_1. \tag{4}$$

First note that $\overline{w}$ must be in $C_1(t)$, since otherwise t would be in W_{uv}, which is empty. Recall that W_{uv} is the quarter-circle in $C_i(p)$ that is centred at p with q on its boundary contained in $C_i(p)$. Therefore we have $\|t\overline{u}\|_1 > \|t\overline{w}\|_1$. The same reasoning shows that $\|t\overline{u'}\|_1 > \|t\overline{w'}\|_1$.

Since u' comes after u on the routing path, we know that it is inside the square centred at t with u on its boundary by Lemma 2. The point u' must be inside the intersection of this square with $C_1(t)$, and below d. If $\|\overline{v}t\|_1 < \|\overline{u'}t\|_1$, then u' would either have to be inside $T(u,d)$ or W_{uv}. Both of these two regions are empty, so we must have $\|\overline{v}t\|_1 \geq \|\overline{u'}t\|_1$. See Figure 4. ◀

We can use this lemma and the fact that greedy edges can only originate in two cones of t to bound the total length of the greedy edges.

▶ **Lemma 8.** *The total L_2 length of the greedy edges is at most $8\|st\|_\infty$.*

Proof. Let $G_1, \ldots, G_4$ be the four sets of greedy edges defined above. The total length of the footprints in one such set cannot be more than $\sqrt{2}\|st\|_\infty$, since that is the length of d that lies inside the intersection of one of the cones of t with the square centred at t with s on its boundary. Therefore,

$$\sum_{uv \in G} \|uv\|_2 = \sum_{i=1}^{4} \sum_{uv \in G_i} \|uv\|_2 = \sum_{i=1}^{4} \sum_{uv \in G_i} \sqrt{2}\|\overline{uu}\|_2 \leq \sum_{i=1}^{4} 2\|st\|_\infty = 8\|st\|_\infty. \quad ◀$$

Now that we have a bound on the length of the greedy edges in terms of $\|st\|_\infty$, we can combine that with our bound on the length of the sweeping edges to get a bound on the total length of the path, and therefore the routing ratio.

▶ **Theorem 9.** *The routing ratio of the greedy-sweep algorithm is at most* $17 + 9/\sqrt{2}$*.*

3.1 Removing the diagonal bit

In the greedy-sweep algorithm, one bit of memory is required to remember the choice of diagonal. Removing the single bit of memory just requires us to fix a diagonal ahead of time, without knowledge of s and t. We will choose $d_-(t)$ as our diagonal. The proof of Lemma 2 does not depend on our choice of d at all, so the routing algorithm will still terminate with this modification.

A few changes to the analysis have to be made, however. The initial height $h(s)$ can now be larger. The height of s for the one bit greedy-sweep is at most $\|st\|_\infty$, because we chose d to be whichever diagonal of t is closer to s. Now, however, since we fix the diagonal ahead of time the height of s is at most $2\|st\|_\infty$. The proofs of Lemmas 6 and 8 do not depend on the choice of diagonal, and so they remain unchanged. Notice how increasing $h(s)$ beyond $\|st\|_\infty$ also increases the distance $\|st\|_2$. This results in the routing ratio being a unimodal function of $\|st\|_\infty$, where at a certain point increasing $h(s)$ actually decreases the routing ratio since $\|st\|_\infty$ increases too quickly. The routing ratio will increase slightly, from $17 + 9/\sqrt{2} \approx 23.36$ to $\sqrt{331 + 154\sqrt{2}} \approx 23.43$.

▶ **Theorem 10.** *The memoryless version of the greedy-sweep algorithm has a routing ratio of at most* $\sqrt{331 + 154\sqrt{2}}$*.*

Proof. Assume without loss of generality that s lies on the left edge of the square centred at t with s on the boundary. By Lemmas 6 and 8 we know that the length of the routing path is at most $\alpha h(s) + (8\alpha + 8)\|st\|_\infty$, where $\alpha = 1 + \frac{1}{\sqrt{2}}$. If $h(s) \leq \|st\|_\infty$ then we can proceed as we did for the proof of Theorem 9.

If $h(s) > \|st\|_\infty$, then let θ be the angle made by the segment st with the ray $R_2(t)$ and let ρ be the routing ratio. Notice that $0 < \theta < \frac{\pi}{4}$. Then

$$\begin{aligned}\rho &\leq \frac{\alpha h(s) + (8\alpha + 8)\|st\|_\infty}{\|st\|_2} \\ &= \alpha \sin\theta + (9\alpha + 8)\cos\theta.\end{aligned}$$

This is maximized when $\theta = \arctan(\alpha/(9\alpha + 8))$ with a value of $\rho = \sqrt{331 + 154\sqrt{2}}$. ◀

4 Spanning ratio

We now turn our attention to the undirected Yao-4 graph. Let s and t be two vertices of a $\vec{Y}_4$ graph, and let $\mathcal{P}$ be the path from s to t constructed by the greedy-sweep algorithm. Notice that this is also a path in the undirected Y_4 graph. This already gives an upper bound of 23.36 on the spanning ratio of the undirected Y_4 graph. In this section, we will improve this upper bound to 16.95.

First, define $P_i(p)$ to be the path constructed by starting at p and following edges in cone i repeatedly, until a point is reached that has no neighbour in cone i. Also, given two points p and q define $R(p, q)$ to be the axis-aligned rectangle that has p and q at opposite corners. The side lengths of this rectangle are $d_x(p, q)$ and $d_y(p, q)$. Define $SS(p, q) = \min\{d_x(p, q), d_y(p, q)\}$ and $LS(p, q) = \max\{d_x(p, q), d_y(p, q)\}$.

Assume without loss of generality that s is in $C_1(t)$, and that it is below $d_-(t)$. In this situation, greedy edges can only originate in $C_1(t)$ and in $C_3(t)$, not $C_0(t)$ or $C_2(t)$. The key point of this section is that if any edge of $\mathcal{P}$ originates in $C_3(t)$, then $\mathcal{P}$ must intersect at least one of $P_0(t)$ and $P_2(t)$. We will show that if two edges of a Y_4 graph intersect, then there is a short path between the endpoints of the two edges. This means we can take a "shortcut" and only use the subpath of $\mathcal{P}$ before its intersection with $P_0(t)$ or $P_2(t)$.

Before proving the main result of this section, we will characterize the possible intersections in a Y_4 graph. First, some definitions. An edge pq is called an i-edge if q is in $C_i(p)$. If pq is an i-edge and uv is an $(i \pm 1)$-edge, then we say that pq and uv are in adjacent cones. If uv is an $(i + 2)$-edge, then we say that they are in opposite cones. Note that an i-edge is the same thing as an $(i \bmod 4)$-edge.

▶ **Lemma 11.** *Let pq and uv be i-edges such that p is not in $C_i(u)$ and u is not in $C_i(p)$. Then pq and uv cannot intersect except if $q = v$.*

The previous lemma implies that edges in the same cone can only intersect at their endpoints. We will characterize intersections of edges in opposite or adjacent cones into two categories: short side and long side crossings. See Figure 5.

If pq and uv are in opposite cones and pq intersects the short side of $R(p, u)$, then we say that pq *short side crosses* uv. Notice that two edges in opposite cones do not have to actually intersect for us to say that they short side cross. If the edges are in adjacent cones and pq intersects the short side of $R(p, v)$, then we say that pq short side crosses uv. If it intersects the long side of $R(p, v)$, then we say that pq *long side crosses* uv. Edges in adjacent cones must intersect if they short or long side cross. If the edges are in adjacent cones and $q = v$, then we will consider the intersection to be a short side crossing.

If pq and uv are in opposite cones and intersect, then we always consider this a short side crossing since one of the edges will short side cross $R(p, u)$, as the next lemma shows.

▶ **Lemma 12.** *Let pq and uv be edges in opposite cones that intersect. Then either pq short side crosses uv, or uv short side crosses pq.*

Note the slight difference in definitions for opposite and adjacent cones. If pq short side crosses uv and they are in adjacent cones, then pq intersects the short side of $R(p, v)$. But if they are in opposite cones, then pq intersects the short side of $R(p, u)$. We will call the second point (other than p) that defines this rectangle the *visible vertex*, so that we can say pq short side crosses an edge with visible vertex u.

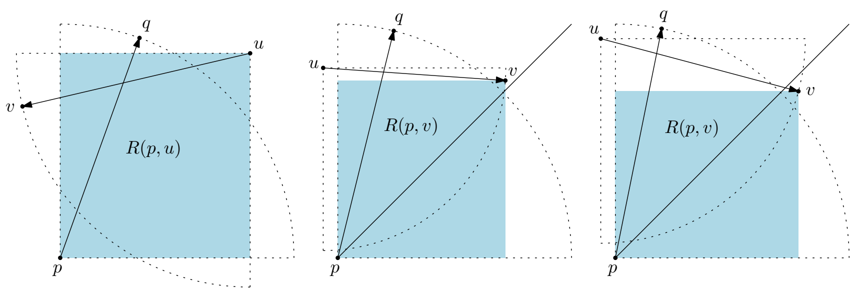

Figure 5 Three different crossings. From left to right: pq opposite cone short side crosses uv with visible vertex u, pq adjacent cone short side crosses uv with visible vertex v, and pq long side crosses uv.

4.1 Short side crossings

In this section we will prove that if an edge short side crosses another edge, then we can find a short undirected path between them. Long side crossings will be considered in the next section. The following two lemmas are simply geometric facts that will be needed to prove the main result of this section.

▶ **Lemma 13.** *Let p and q be two distinct points. Then $\|pq\|_2 \leq LS(p,q) + (\sqrt{2}-1)SS(p,q)$.*

Proof. Consider a right triangle with legs $LS(p,q)$ and $SS(p,q)$. Let θ be the angle adjacent to the hypotenuse and the leg with length $LS(p,q)$. We have $SS(p,q) = \|pq\|_2 \sin\theta$ and $LS(p,q) = \|pq\|_2 \cos\theta$.

$$(\sqrt{2}-1)SS(p,q) + LS(p,q) = \|pq\|_2((\sqrt{2}-1)\sin\theta + \cos\theta). \tag{5}$$

The function $(\sqrt{2}-1)\sin\theta + \cos\theta$ is at least 1 on the interval $[0, \pi/4]$. ◀

▶ **Lemma 14.** *Let pq be an edge that short side crosses an edge with visible vertex u. Then*
(1) *$LS(q,u) \leq SS(p,u)$, and*
(2) *$SS(q,u) \leq (\sqrt{2}-1)SS(p,u)$.*

Now we are ready to state the main result of this section.

▶ **Lemma 15.** *Let pq be an edge of a Y_4 graph that short side crosses another edge with visible vertex u. Then there is an undirected path from p to u with length at most $LS(p,u) + (\sqrt{2}+1)SS(p,u)$.*

Proof. The proof is by induction on all pairs of points $p \neq u$ such that there is an edge pq that short side crosses an edge with visible vertex u, ordered by $SS(p,u)$.

Consider one such pair. There are two possibilities. If $q = u$, then there is a path between p and u with length $\|pu\|_2 \leq \|pu\|_1 < LS(p,u) + (\sqrt{2}+1)SS(p,u)$. Otherwise $q \neq u$. Assume without loss of generality that pq is a 0-edge and that pq crosses the top edge of $R(p,u)$. Consider $P_3(q)$. We claim that this path must exit $R(q,u)$ by the right edge. Since u is in $C_3(q)$, the path must intersect some edge of $R(q,u)$.

If pq adjacent short side crosses an edge vu, then vu must be a 3-edge. Therefore $P_3(q)$ cannot intersect vu by Lemma 11, so it must intersect the right edge of $R(q,u)$ because vu is above the bottom edge. If pq opposite short side crosses an edge uv, then $R(p,u)$ must be empty since it is contained in the intersection of W_{pq} and W_{uv}. Let q' be the last point on $P_3(q)$. We have $d_y(p,q') < d_x(q',u)$, so the path $P_3(q)$ must exit $R(q,u)$ to the right.

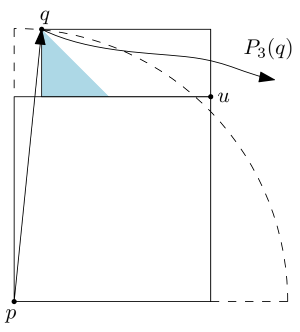

Figure 6 Figure for Lemma 15. Notice that $P_1(u)$ must either intersect pq or $P_3(q)$. If u' is the last vertex on $P_1(u)$ inside $R(q,u)$, then the long side of $R(q,u')$ is horizontal because the shaded region of $R(q,u')$ is contained in W_{pq}.

This implies that $P_1(u)$ must either intersect pq or $P_3(q)$. Both of these will result in a short side crossing. Let u' be the last point on $P_1(u)$ that is inside $R(q,u)$. If $P_1(u)$ intersects $P_3(q)$, there is a short side crossing since the edges will be in opposite cones, and edges in opposite cones that intersect always short side cross. Otherwise if $P_1(u)$ intersects pq, the long side of $R(q,u')$ is horizontal, so we have a short side crossing. Notice that this short side crossing has a smaller short side length than $SS(p,u)$. See Figure 6.

This implies that in the pair p and u such that $SS(p,u)$ is minimized, there is an edge from p to u. So in the base case there is a path between p and u with length at most $\|pu\|_2$.

Now assume for the inductive step that we have a pair of points p and u such that there is an edge pq that short side crosses an edge with visible vertex u, and that for every other such pair p' and u' such that $SS(p',u') < SS(p,u)$, there is a path from p' to u' of length at most $LS(p',u') + (\sqrt{2}+1)SS(p',u')$. If $P_1(u)$ intersects pq, then construct a path in the following manner. By induction, there is a path between q and u' with length at most $LS(q,u') + (\sqrt{2}+1)SS(q,u')$. Concatenate the edge pq, the path between q and u', and the segment of $P_1(u)$ up to u'. The resulting path has length at most

$$\begin{aligned} d_G(p,u) &\leq \|pq\|_2 + LS(q,u') + (\sqrt{2}+1)SS(q,u') + \|uu'\|_1 \\ &\leq \|pq\|_2 + LS(q,u) + (\sqrt{2}+1)SS(q,u). \end{aligned} \tag{6}$$

Now, if $P_1(u)$ intersects $P_3(q)$, then

$$\begin{aligned} d_G(p,u) &\leq \|pq\|_2 + \|qq'\|_1 + LS(q',u') + (\sqrt{2}+1)SS(q',u') + \|uu'\|_1 \\ &\leq \|pq\|_2 + LS(q,u) + (\sqrt{2}+1)SS(q,u). \end{aligned} \tag{7}$$

In both cases we have the same bound on the length of the path. Now, Lemma 13 implies that $\|pq\|_2 \leq LS(p,u) + (\sqrt{2}-1)SS(p,u)$, and Lemma 14 gives bounds on $LS(q,u)$ and $SS(q,u)$. Putting these together finishes off the proof:

$$\begin{aligned} d_G(p,u) &\leq \|pq\|_2 + LS(q,u) + (\sqrt{2}+1)SS(q,u) \\ &\leq LS(p,u) + (\sqrt{2}-1)SS(p,u) + SS(p,u) + (\sqrt{2}+1)(\sqrt{2}-1)SS(p,u) \\ &= LS(p,u) + (\sqrt{2}+1)SS(p,u). \end{aligned}$$ ◀

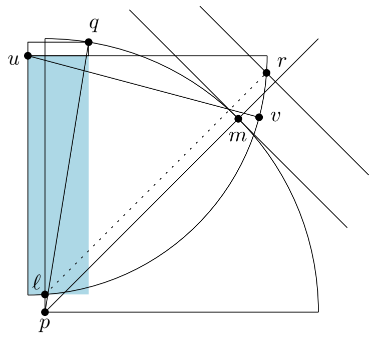

Figure 7 Figure for Lemma 17.

4.2 Long side crossings

Suppose we have an edge pq that long side crosses another edge uv. In this section, we will show that there is a short path from q to u, using the short side crossings from the previous section. Assume without loss of generality that uv is a 3-edge and pq is a 0-edge. We will show that from u and q we can find two paths in opposite cones that must intersect, and by Lemma 12 that intersection must be a short side crossing.

▶ **Lemma 16.** *$P_0(u)$ intersects the top edge of $R(u,q)$.*

Proof. The path $P_0(u)$ cannot intersect the right edge, since then it would have to intersect the edge pq. But pq is a 0-edge, and two 0-edges cannot intersect by Lemma 11. ◀

▶ **Lemma 17.** *$P_2(q)$ intersects the left edge of $R(u,q)$.*

Proof. Consider Figure 7. The point ℓ is directly above p and on the arc of W_{uv}. The point r is the other intersection of the line $d_+(\ell)$ with the arc of W_{uv}.

Every point on the arc of W_{pq} is below $d_-(m)$, including q. The point r is above $d_+(p)$ and on the arc of W_{uv}. This means that r must be above and to the right of m. Therefore q is also below $d_-(r)$.

Since q is above r but below $d_-(r)$, it must be the case that q is to the left of r. In other words, we have $d_x(u,q) < d_x(u,r) = d_x(u,y)$, which is what we wanted to show.

We have shown that $d_y(u,\ell) > d_x(u,q)$. That implies that the rectangle R with u as its upper-left corner, with width $d_x(u,q)$ and height $d_y(u,\ell)$, is taller than it is wide. Notice that R is contained in $W_{pq} \cup W_{uv}$, meaning that it is empty of points. Consider an edge xy on $P_2(q)$ such that x is in $R(u,q)$. If y is below ℓ, then we must have

$$\|xy\|_2 > d_y(x,y) > d_y(x,\ell) > d_y(x,u) + d_x(x,u) > \|xu\|_2,$$

a contradiction since both y and u are in $C_2(x)$. Therefore no edge of $P_2(q)$ can span R in this sense, and $P_2(q)$ must intersect the left edge of $R(u,q)$. ◀

These two lemmas imply the main result of this section.

▶ **Lemma 18.** *Let pq be an edge of a Y_4 graph that long side crosses another edge uv. Then there is an undirected path between u and q with length at most $LS(u,q) + (\sqrt{2}+1)SS(u,q)$.*

4.3 Constructing a path

In this section we describe how to construct a path between two vertices of an undirected Y_4 graph. Let s and t be the endpoints of our path. Let $\mathcal{P}$ be the path from s to t constructed by the greedy sweep algorithm in the directed $\vec{Y_4}$ graph. This is also a path in the undirected Y_4 graph. For the remainder of this section, assume without loss of generality that s is above and to the left of t, and that s is below the diagonal $d_-(t)$.

We consider three cases, based on the observation that if some vertex of $\mathcal{P}$ lies in $C_3(t)$, then $\mathcal{P}$ must intersect at least one of $P_0(t)$ or $P_2(t)$. We will construct a path in a different way for each of the three cases. The cases that we consider then are:

1. The path $\mathcal{P}$ does not intersect $P_0(t) \cup P_2(t)$, except for at t
2. The first intersection of $\mathcal{P}$ with $P_0(t) \cup P_2(t)$ is a short side crossing
3. The first intersection of $\mathcal{P}$ with $P_0(t) \cup P_2(t)$ is a long side crossing

In the first case, the path between s and t is $\mathcal{P}$ itself.

In the second case, let uv be the first edge of $\mathcal{P}$ that intersects $P_i(t)$, where $i \in \{0, 2\}$. If u is in $C_i(t)$, then uv must be an $(i+2)$-edge, so the edge pq that it intersects is in the opposite cone. Therefore by Lemma 15 there is a path between u and p with length at most $LS(p, u) + (\sqrt{2} + 1)SS(p, u)$. Otherwise, if u is in $C_1(t)$, then uv must be a 3-edge and we have edges in adjacent cones that intersect. In this case we must have pq short side crossing uv, with visible vertex v. By Lemma 15 there is a path between v and p with length at most $LS(p, v) + (\sqrt{2} + 1)SS(p, v)$. In either case concatenating the subpath of $\mathcal{P}$ from s to the visible vertex (u in the case of an opposite cone short side crossing, and v in the case of an adjacent cone short side crossing), the path from the visible vertex to p of Lemma 15, and the subpath of $P_i(t)$ from t to p (in reverse) gives a path between s and t.

The third case is similar to the second. Let uv be the first edge of $\mathcal{P}$ that intersects $P_i(t)$, where $i \in \{0, 2\}$. In this case pq long side crosses uv, meaning they must be in adjacent cones, and uv is a 3-edge, and u is in $C_1(t)$. By Lemma 18 there is a path between u and q. Concatenate the subpath of $\mathcal{P}$ from s to u, the path from u to q, and the subpath of $P_i(t)$ from t to q (in reverse).

▶ **Theorem 19.** *Let P be a set of points. For any two points s and t in P, there is a path in $Y_4(P)$ with length at most $(13 + 5/\sqrt{2})\|st\|_2$.*

Proof. The path is constructed as previously described. We will bound the length for each case separately.

Case 1. Since no edge of P originates in $C_3(t)$, we know that any greedy edge of P will have to originate in $C_1(t)$. We can modify the proof of Lemma 8 using this fact. Now there are only two subsets of greedy edges to consider, depending on whether they originate above or below the diagonal. This results in a total length of $4\|st\|_\infty$ for the greedy edges. The proof of Lemma 6 does not need to change at all, giving a total length for P of

$$\begin{aligned}
\sum_{uv \in S} \|uv\|_2 + \sum_{uv \in G} \|uv\|_2 &\leq \left(1 + \frac{1}{\sqrt{2}}\right)\left(h(s) + \sum_{uv \in G} \|uv\|_2\right) + \sum_{uv \in G} \|uv\|_2 \\
&\leq \left(1 + \frac{1}{\sqrt{2}}\right)\left(\|st\|_\infty + 4\|st\|_\infty\right) + 4\|st\|_\infty \\
&\leq \left(9 + \frac{5}{\sqrt{2}}\right)\|st\|_2.
\end{aligned}$$

Case 2. Let u be the vertex of P that is the visible vertex of the first intersection of P with $P_i(t)$. Let P' be the subpath of P from s to u. Notice P' does not have any edges originating in $C_3(t)$. Therefore the total length of the greedy edges on this subpath is at most $4\|st\|_\infty$ as in case 1. Now we modify the proof of Lemma 6 by noting that

$$\sum_{pq\in D'}(h(p)-h(q)) = h(s)-h(u)+\sum_{pq\in I'}(h(q)-h(p)) \tag{8}$$

where D' and I' are the sets of edges of P' where the height decreases and increases, respectively, as in the proof of Lemma 6. Using that fact we see that the length of P' is at most $(8+4/\sqrt{2})\|st\|_\infty + (1+1/\sqrt{2})(h(s)-h(u))$.

Next, we know the length of the path between u and p by Lemma 15: at most $(\sqrt{2}+1)SS(u,p)+LS(u,p)$. And finally the length of the subpath of $P_i(t)$ is at most $\|pt\|_1$. Notice that p is inside $R(t,u)$, so we have

$$\begin{aligned}(\sqrt{2}+1)SS(u,p)+LS(u,p)+\|pt\|_1 &= (\sqrt{2}+1)SS(u,p)+LS(u,p)+SS(p,t)+LS(p,t)\\ &\le (\sqrt{2}+1)SS(t,u)+LS(t,u).\end{aligned}$$

Now, we have $SS(t,u)+LS(t,u)=h(u)$. Furthermore, $h(u)/2 \le LS(t,u) \le h(u)$. Therefore,

$$\begin{aligned}(\sqrt{2}+1)SS(t,u)+LS(t,u) &= (\sqrt{2}+1)(h(u)-LS(t,u))+LS(t,u)\\ &= (\sqrt{2}+1)h(u)-\sqrt{2}LS(t,u)\\ &\le (\sqrt{2}+1)h(u)-\frac{\sqrt{2}}{2}h(u)\\ &= \left(1+\frac{1}{\sqrt{2}}\right)h(u).\end{aligned}$$

Adding this to the length of P', we see that the length of the path between s and t is at most

$$\begin{aligned}&\left(8+\frac{4}{\sqrt{2}}\right)\|st\|_\infty+\left(1+\frac{1}{\sqrt{2}}\right)(h(s)-h(u))+\left(1+\frac{1}{\sqrt{2}}\right)h(u)\\ &\quad= \left(8+\frac{4}{\sqrt{2}}\right)\|st\|_\infty+\left(1+\frac{1}{\sqrt{2}}\right)h(s)\\ &\quad\le \left(9+\frac{5}{\sqrt{2}}\right)\|st\|_\infty\\ &\quad\le \left(9+\frac{5}{\sqrt{2}}\right)\|st\|_2.\end{aligned}$$

Interestingly, this is the same bound as in case 1.

Case 3. Just like in case 2, the subpath P' from s to u has length at most $4\|st\|_\infty + (1+1/\sqrt{2})(4\|st\|_\infty + h(s)-h(u)) \le (9+5/\sqrt{2})\|st\|_\infty$. By Lemma 18, the length of the path between u and q is at most $LS(u,q)+(\sqrt{2}+1)SS(u,q)$. Finally, the subpath of $P_i(t)$ from t to q has length at most $\|tq\|_1$.

We will show that $LS(u,q)\le d_y(u,p)$ and $SS(u,q)\le(\sqrt{2}-1)d_y(u,p)$. That will imply that the length of the subpath between u and q has length at most $d_y(u,p)+(\sqrt{2}+1)(\sqrt{2}-1)d_y(u,p) = 2d_y(u,p) \le 2d_y(u,t) \le 2\|st\|_\infty$. We also have $\|qt\|_1 \le 2\|qt\|_\infty \le 2\|st\|_\infty$. So the total length of the subpath between u and t is at most $4\|st\|_\infty$. Adding that to the length of P' gives a total of $(13+5/\sqrt{2})\|st\|_\infty$.

Consider Figure 8. Without loss of generality, we assume that pq is an edge on $P_0(t)$. Let $x=d_x(u,p)$ and $y=d_y(u,p)$. Let u' be the point directly above p such that $d_y(u',p)=y$.

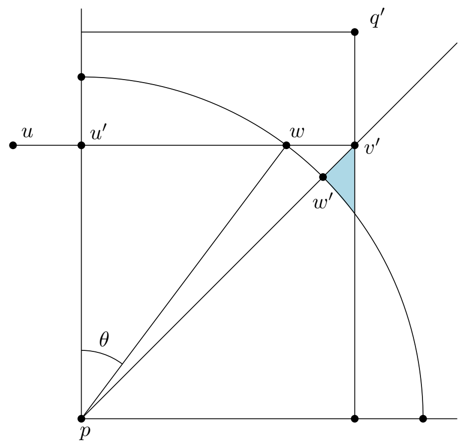

Figure 8 Case 3 of Theorem 19.

Let v' be the point of the bisector of $C_0(p)$ such that $d_x(v', p) = d_y(v', p) = y$. Let w be the point between u' and v' such that $d_x(w, v') = x$. Let w' be the point on the bisector of $C_0(t)$ such that $\|pw'\|_2 = \|pw\|_2$. Finally, let q' be the point above v' such that $d_y(p, q') = \|pv'\|_2$.

First, some observations. The point v must be to the left of v' since $\|uv'\|_2 = \|up\|_1 \geq \|uv\|_2$. The point q must be inside $R(u', q')$, since q is above u and right of p, but if it were outside of the rectangle then it would contain v' and the point v could not exist, because W_{pq} is empty of points.

We will show that $LS(u, q) \leq LS(u', q')$ and $SS(u, q) \leq SS(u', q')$. Then, since $LS(u', q') = y$ and $SS(u', q') = d_y(u', q') = (\sqrt{2} - 1)y$, we will have completed the proof.

First since q is inside $R(u', q')$ we must have $d_y(u, q) \leq d_y(u', q')$. Next we show that $d_x(u, q) \leq d_x(u', q')$. To do this we will show that q is to the left of w. Then we would have $d_x(u, q) \leq d_x(u, w) = d_x(u', q')$. We know that $\|uv\|_2 < \|up\|_2$, so if we can show that $\|up\|_2 < \|uw'\|_2$ then we will know that v cannot lie in the shaded region of Figure 8. That would mean that W_{pq} is inside W_{pw}, so $\|pq\|_2 < \|pw\|_2$, and since q is above w this must mean that q is left of w.

Now we prove that $\|up\|_2 < \|uw'\|_2$. Let $r = d_x(p, w') = d_y(p, w')$. Notice that $\|up\|_2^2 = x^2 + y^2$ and $\|uw'\|_2^2 = (x + r)^2 + (y - r)^2$. If we can show that the inequality $x^2 + y^2 < (x + r)^2 + (y - r)^2$ holds, we will be done. The inequality can be simplified to $x + r > y$. This holds since w is left of w', so $x + r = d_x(u, w') > d_x(u, w) = y$.

Therefore $\|uw'\|_2 > \|up\|_2$, which implies that q is to the left of w, which implies that $d_x(u, q) \leq d_x(u', q')$. This together with the fact that $d_y(u, q) \leq d_y(u', q')$ means we must have $LS(u, q) \leq LS(u', q') = d_y(u, p)$ and $SS(u, q) \leq SS(u', q') = (\sqrt{2} - 1)d_y(u, p)$. As we have previously shown, this means the length of the path between s and t has length at most $(13 + 5/\sqrt{2})\|st\|_2 \approx 16.54\|st\|_2$. Notice that the bound in this case is greater than the bound for the other two cases, by exactly $4\|st\|_2$. ◀

5 Conclusion

We have presented a local routing algorithm for the directed $\vec{Y}_4$ graph, the first such routing algorithm for this class of graphs. The routing ratio of this algorithm is $17 + 9/\sqrt{2} \approx 23.36$. The algorithm requires one bit of memory, and we showed that this can be dispensed with at the cost of increasing the routing ratio to $\sqrt{331 + 154\sqrt{2}} \approx 23.42$. Our result also lowers the best-known upper bound on the spanning ratio of the directed $\vec{Y}_4$ graph from 54.82 to

23.36. We also used the routing algorithm to bound the spanning ratio of the undirected Y_4 graph to be at most 16.54. To do so we showed that if two edges of a Y_4 graph intersect, then there must be a short path between the endpoints of these edges.

References

1 B. E. Flinchbaugh and L. K. Jones. Strong connectivity in directional nearest-neighbour graphs. *SIAM Journal on Algebraic Discrete Methods*, 2(4):461–463, 1981. `doi:10.1137/0602049`.

2 Luis Barba, Prosenjit Bose, Mirela Damian, Rolf Fagerberg, Wah Loon Keng, Joseph O'Rourke, André van Renssen, Perouz Taslakian, Sander Verdonschot, and Ge Xia. New and improved spanning ratios for Yao graphs. In *Proceedings of the Thirtieth Annual Symposium on Computational Geometry*, SOCG'14, pages 30–39, New York, NY, USA, 2014. Association for Computing Machinery. `doi:10.1145/2582112.2582143`.

3 Prosenjit Bose, Jean-Lou De Carufel, Darryl Hill, and Michiel Smid. On the spanning and routing ratio of theta-four. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2361–2370. SIAM, 2019. `doi:10.1137/1.9781611975482.144`.

4 Kenneth L. Clarkson. Approximation algorithms for shortest path motion planning. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, pages 56–65, 1987. `doi:10.1145/28395.28402`.

5 Nawar M El Molla. *Yao spanners for wireless ad hoc networks*. Villanova University, 2009.

6 Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993. `doi:10.1007/BF02189308`.

7 William Michael Zoltan Kalnay. *Routing Ratio of the Directed Yao-6 Graphs*. Carleton University, 2023.

8 J. Mark Keil. Approximating the complete Euclidean graph. In *1st Scandinavian Workshop on Algorithm Theory*, volume 318 of *Lecture Nodes in Computer Science*, pages 208–213, 1988. `doi:10.1007/3-540-19487-8_23`.

9 Mirela Damian and Kristin Raudonis. Yao graphs span theta graphs. In *Combinatorial Optimization and Applications COCOA 2010*, volume 6509 of *Lecture Nodes in Computer Science*, pages 181–194, 2010. `doi:10.1007/978-3-642-17461-2_15`.

10 Mirela Damian and Naresh Nelavalli. Improved bounds on the stretch factor of Y_4. *Computational Geometry*, 62:14–24, 2017. `doi:j.comgeo.2016.12.001`.

11 Prosenjit Bose, Anil Maheshwari, Giri Narasimhan, Michiel Smid, and Norbert Zeh. Approximating geometric bottleneck shortest paths. *Computational Geometry*, 29(3):233–249, 2004. `doi:10.1016/j.comgeo.2004.04.003`.

12 Prosenjit Bose, Mirela Damian, Karim Douïeb, Joseph O'Rourke, Ben Seamone, Michiel Smid, and Stefanie Wuhrer. $\pi/2$-angle Yao graphs are spanners. In *Algorithms and Computation – 21st International Symposium, ISAAC 2010*, volume 6507 of *Lecture Nodes in Computer Science*, pages 446–457, 2010. `doi:10.1007/978-3-642-17514-5_38`.

13 Andrew Chi-Chih Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM Journal on Computing*, 11(4):721–736, 1982. `doi:10.1137/0211059`.