Local Routing in Spanners Based on WSPDs

Frédérik Paradis

Thesis submitted in partial fulfilment of the requirements for the
Master's degree of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Electrical Engineering and Computer Science
University of Ottawa

Abstract

The well-separated pair decomposition (WSPD) of the complete Euclidean graph defined on points in $\mathbb{R}^2$, introduced by Callahan and Kosaraju [JACM, 42 (1): 67-90, 1995], is a technique for partitioning the edges of the complete graph based on length into a linear number of sets. Among the many different applications of WSPDs, Callahan and Kosaraju proved that the sparse subgraph that results by selecting an arbitrary edge from each set (called WSPD-spanner) is a $1+8/(s-4)$-spanner, where $s > 4$ is the separation ratio used for partitioning the edges.

Although competitive local-routing strategies exist for various spanners such as Yao-graphs, Θ-graphs, and variants of Delaunay graphs, few local-routing strategies are known for any WSPD-spanner. Our main contribution is a local-routing algorithm with a near-optimal competitive routing ratio of $1 + O(1/s)$ on a WSPD-spanner.

Specifically, using Callahan and Kosaraju's fair split-tree, we show how to build a WSPD-spanner with spanning ratio $1+4/s+4/(s-2)$ which is a slight improvement over $1 + 8/(s - 4)$. We then present a 2-local and a 1-local routing algorithm on this spanner with competitive routing ratios of $1 + 6/(s - 2) + 4/s$ and $1 + 8/(s - 2) + 4/s + 8/s^2$, respectively. Moreover, we prove that there exists a point set for which our WSPD-spanner has a spanning ratio of at least $1 + 8/s$, thereby proving the near-optimality of its spanning ratio and the near-optimality of the routing ratio of both our routing algorithms.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

A fundamental problem in networking is the routing of a message from one vertex to another in a graph. Because network resources are limited, it is often desirable that routing algorithms use as little memory as possible. At one extreme in this direction are *local* routing algorithms where the routing algorithm must choose the next vertex to forward a message to based solely on knowledge of the destination vertex, the current vertex and some information about all vertices directly connected to the current vertex. When a local routing algorithm is not possible, it is still desirable that a routing algorithm use as little memory as possible.

In many settings, it is natural to model a network as a *geometric graph*, that is, a graph whose vertices are points and each edge is a line segment whose weight is the Euclidean distance between its two endpoints. For example, geometric routing algorithms are important in wireless sensor networks (see [21] for a survey of the area) since routing strategies can take advantage of the fact that nodes in these networks have physical locations that can be used to help guide a packet to its destination.

1.1 Motivation

A geometric routing algorithm is said to be *competitive* if the length of all paths produced by the routing algorithm is not more than a constant times the Euclidean distance between its endpoints. This smallest such constant is called the *routing ratio*. In order to find a competitive path (i.e. a path that satisfies the routing ratio) between any two vertices of a graph, such a path must first exist. Graphs that meet this criterion are called (geometric) spanners. Formally, given a geometric graph G, the distance, $d_G(u, v)$, between two vertices u and v in G is the sum of the weights of the edges in the shortest path between u and v in G. The graph G is a *t-spanner* if there exists a $t \geq 1$ such that for all pairs of vertices u and v in G, $d_G(u, v) \leq t \cdot |uv|$. Here $|uv|$ denotes the Euclidean distance between u and v. The smallest value t for which G is a t-spanner is the *spanning ratio* or *stretch factor* of G. A family of graphs that are t-spanners, for some fixed constant t, are often referred to as simply *spanners*. Spanners have been extensively studied—for a detailed overview of results on geometric spanners, see the book by Narasimhan and Smid [22].

Geometric spanners tend to fall into three categories: (i) Long-known geometric graphs that happen to be spanners, such as Delaunay triangulations; (ii) cone-based constructions, such as Keil's θ-graphs [20]; and (iii) well-separated pair decomposition(WSPD) based constructions introduced by Callaghan and Kosaraju [14]. Note that graphs in the first category have fixed worst-case spanning ratios bounded away from 1. Constructions in the second and third categories are designed for a given parameter. They can achieve spanning ratios arbitrarily close to 1 by choosing arbitrarily small values for this parameter. Significant work has gone into finding competitive local and low-memory routing algorithms for graphs in the first category, including Delaunay graphs (classical–, L_1–, L_∞–, TD–, and generalized convex Delaunay triangulations) [6, 7, 10, 11, 15]. In most cases, proving tight spanning ratios and routing ratios for graphs in this category is difficult. For example, even the exact spanning ratio of the Delaunay triangulation is unknown, despite over 30 years of study [9, 17, 20, 23].

For the second category—cone-based spanners—competitive local routing algorithms are usually trivial. These spanners are designed so that greedy choices produce paths of low stretch. Still, for certain cone-based spanners, there have been some refined results on competitive routing algorithms that produce exceptionally low competitive ratios. For example, Bose *et al* [10] present a routing algorithm for the TD-Delaunay triangulation (which is equivalent to the Half-θ_6-graph) with a competitive ratio of 2.887. They prove that this is optimal, thereby proving a separation between the routing ratio and the spanning ratio of a graph since the spanning ratio of the TD-Delaunay triangulation is 2 [16].

1.2 Problem Statement

In this thesis, we consider routing algorithms for the third category: WSPD-based spanners. Intuitively, a WSPD of a pointset is a partition of the edges of the complete geometric graph (on that pointset) such that all edges in the same partition are approximately of equal length.[1] Since its introduction by Callahan and Kosaraju [14], the WSPD and WSPD-based spanners have found a plethora of applications in solving distance problems [22]. The main difficulty about local routing in these spanners stems from the fact that WSPD-spanners are based on WSPDs that are built globally and capture global distance properties of the given pointset. As such WSPD-spanners pose a challenge in designing local routing strategies.

WSPDs have been used before as an aid to routing in unit-disk graphs by Kaplan et al. [19]. Their scheme applies to our setting when the unit distance is the diameter of the point set. However, in that case, they route on an ϵ-net of the point set. Therefore, they are not routing purely on a WSPD-spanner of the complete graph of the point set but they are routing on the subset of the points that forms the ϵ-net. In the case where the unit disk graph has diameter at least 2, their routing scheme requires a header of $O(\log n \log D)$ bits, where D is the diameter. It also requires routing tables of size $O(\epsilon^{-5} \log^2 n \log^2 D)$ bits per vertex and, therefore, the total size of the routing tables is $O(n\epsilon^{-5} \log^2 n \log^2 D)$ bits. Their routing ratio is $1 + \epsilon$ where $\epsilon = (\alpha/s) \log |pq|$ (or $s = (\alpha/\epsilon) \log |pq|$) with $\alpha \geq 192$ and $|pq| \leq D \leq n$. Note that our scheme is slightly different and therefore incomparable since it is routing on a WSPD-spanner.

1.3 Contribution

Given a pointset and a separation ratio s, a WSPD with separation ratio s is (typically) not unique. Callahan and Kosaraju's original construction of a WSPD is based on fair split-trees and it computes a WSPD containing a linear number of edge partitions [14]. From this WSPD, we show how to construct a WSPD-spanner that facilitates local routing by selecting a well-chosen edge from each partition rather than

[1] See the next section for the formal definition.

picking an arbitrary edge (see Section 3.2). As a side benefit, our WSPD-spanner has a slightly improved spanning ratio, $1+4/s+4/(s-2)$, over the original one, $1+8/(s-4)$. This improvement stems from the additional properties of our well-chosen edges. On this WSPD-spanner, we present a 2-local and a 1-local routing algorithm with competitive routing ratios of $1+6/(s-2)+4/s$ and $1+8/(s-2)+4/s+8/s^2$, respectively (see Sections 3.3 and 3.4). A routing algorithm on a graph G is *k-local* for $k \geq 1$ if each vertex v of G stores information about vertices that are at a hop distance of at most k from v. The hop distance between two vertices p and q is k if the minimum number of edges to traverse in order to reach q from p is k. Our local routing algorithms do not use a header. Our 2-local and 1-local routing algorithms require routing tables of total size $O(s^2n^2B)$ and $O(s^2nB)$ bits, respectively, where B is the maximum number of bits to store a bounding box. Ideally, one would like the routing ratio to be identical to the spanning ratio, however, this is rarely the case when routing locally since an adversary can often force an algorithm to stray from the actual shortest path. Finally, we prove a lower bound of $1+8/s$ on the spanning ratio of our WSPD-spanner, thereby proving the near-optimality of the spanning ratio of our WSPD-spanner and the near-optimality of the routing ratios of both our routing algorithms.

1.4 Organization of the Thesis

The rest of the thesis is organized as follows. Chapter 2 presents a literature review of the main results on the well-separated pair decomposition. We begin by stating results about on WSPD-spanners. This is then followed by results about applications of the well-separated pair decomposition. Chapter 3 presents our results together with proofs. Finally, Chapter 4 gives a summary of our results and directions about future research.

Chapter 2

Background

In this chapter, we present a literature review related to the subject of this thesis. Section 2.1 gives some definitions that will be useful throughout this thesis. Section 2.2 discuss known results on WSPD-spanners. Finally, Section 2.3 discusses the well-separated pair decomposition and its application to geometric problems.

2.1 Definitions

A *network* (or *graph*) $G = (V, E)$ is represented by its set of vertices V and its set of edges E. A *local routing algorithm* in a network G routes a message (or packet) by finding a path from a vertex $p \in V$ to a vertex $q \in V$ by making a sequence of local decisions. A decision from a vertex v is said to be *local* if the choice of the next vertex to forward the message to depends only on information accessible from v. The goal is to design a local routing algorithm that uses the smallest amount of local information.

Formally, a 1-local routing algorithm as defined in [8] is a function $f : V \times V \times V \times \mathcal{P}(V) \to V$, where $\mathcal{P}(V)$ is the power set of V. The arguments of the function $f(v, p, q, \mathcal{N}(v)) = w$ are the current vertex v in the path, the source p of the path, the destination q of the path, the set of neighbors $\mathcal{N}(v)$ of v, and the next vertex w on the current path. Our definition of 1-local deviates slightly since in addition to the coordinates of the neighbors in $\mathcal{N}(v)$, we require some additional information to be stored based on the construction of the WSPD. In Chapter 3, we define precisely what additional information is stored. For the remainder of the thesis, when we refer to 1-local, we will refer to this enhanced definition. Let $k \geq 1$ be an integer. We say that a local routing algorithm is *k-local* if each vertex v has access to the graph $G_k(v)$ which consists of the subgraph of G induced by all vertices at hop-distance at most k from v.

In this thesis, we consider *geometric networks.* A geometric network is a network where the vertices are points in the plane and the edges are straight line segments. We consider the Euclidean length of an edge to be its weight. When routing in a geometric network, the length of the path found by the algorithm is the sum of the lengths of all the edges of the path. Let P_{pq} be the path produced by a routing algorithm from p to q. The *routing ratio* is defined as $\max_{x,y \in V} \frac{|P_{xy}|}{|xy|}$, where $|P_{xy}|$ is the length of P_{xy} and $|xy|$ is the Euclidean distance between x and y. A routing algorithm is *competitive* if it has a constant upper bound on its routing ratio. The *spanning ratio* of a geometric graph is defined as $\max_{x,y \in V} \frac{|SP_{xy}|}{|xy|}$, where SP_{xy} is the shortest path from x to y in the graph. A graph is a *t-spanner* if and only if its spanning ratio is at most t for some $t > 1$. In this thesis, when we refer to a graph as a *spanner*, we mean that t is a constant.

2.2 WSPD-Spanner

The *well-separated pair decomposition (WSPD)* of the complete Euclidean graph defined on points in $\mathbb{R}^d$ (where $d \geq 1$ is an integer), introduced by Callahan and Kosaraju [14], is a technique for partitioning the edges of the complete graph based on length into a linear number of sets (see formal definition in Section 2.3). In Callahan and Kosaraju's spanner construction [12], an arbitrary edge is selected from each partition in the WSPD of the point set. The endpoints of the edge are called *representatives* of their respective point set in the pair. They proved that the resulting graph is a spanner with a spanning ratio of at most $1 + 8/(s - 4)$, where s is called the separation ratio of the WSPD. It is important to note that since Callahan and Kosaraju [14] proved that the number of pairs is linear in $O(s^d n)$, the number of edges of the spanner is also linear.

WSPDs have been used before as an aid to routing in a slightly different setting than ours. Kaplan et al. [19] used WSPDs to locally route on unit-disk graphs. Their routing scheme requires a header of $O(\log n \log D)$ bits, where D is the diameter. It also requires routing tables of size $O(\epsilon^{-5} \log^2 n \log^2 D)$ bits per vertex and, therefore, the total size of the routing tables is $O(n\epsilon^{-5} \log^2 n \log^2 D)$ bits. Their routing ratio is $1 + \epsilon$ where $\epsilon = (\alpha/s) \log |pq|$ (or $s = (\alpha/\epsilon) \log |pq|$) with $\alpha \geq 192$ and $|pq| \leq D \leq n$.

2.3 Well-Separated Pair Decomposition

Two point sets A and B are *well-separated* if and only if there are two circles with the same radius ρ respectively enclosing A and B and the minimum distance between the circles is $s\rho$, where $s > 0$ is called the separation ratio.

Definition 1 (Well-Separated Pair Decomposition (WSPD))**.** *The well-separated pair decomposition (WSPD) of a point set $S \subseteq \mathbb{R}^d$ is a set of well-separated pairs $\{\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}\}$ such that for any distinct points p and q in S, there is a unique pair $\{A_i, B_i\}$, $1 \leq i \leq m$, such that $p \in A_i$ and $q \in B_i$, or $p \in B_i$ and $q \in A_i$.*

Originally, Callahan and Kosaraju [14] introduced the well-separated pair decomposition to compute the *potential fields* of particles in particle simulations. With a naive algorithm, the potential fields of particles (modeled as points) can be computed in $O(n^2)$ time. Using a WSPD of the point set, they showed how to compute it in $O(n)$ time. In the same paper, they also used the WSPD to compute the *k-nearest neighbors* of a point in $O(kn)$ time. In another paper [13], they showed how to dynamically insert and delete points in a WSPD in $O(\log^2 n)$ time. They also showed how to maintain the closest pair of points in a point set in $O(\log^2 n)$ using a dynamic WSPD.

As mentioned in Section 2.2, Callahan and Kosaraju [12] also showed how to construct a spanner of a point set with a WSPD. Recall that, in this construction, an edge is added between the representatives of the pairs of the WSPD. Notice that these representatives are chosen arbitrarily. By choosing them more carefully, it is possible to obtain spanners with additional interesting graph properties. The properties most addressed in the literature are bounded diameter, bounded degree, and low weight. The *diameter* of a graph is the maximum number of edges in the shortest path between any two vertices in the graph. The *degree* of a vertex is its number of neighbors. The *weight* $w(G)$ of a graph G is the sum of the Euclidean lengths of the edges of the graph. We usually compare the weight of a spanner to the weight of its *minimum spanning tree* (MST) which is noted $w(MST)$. The MST of a graph is a connected spanning subgraph of minimum weight.

In this thesis, we say that a spanner is *WSPD-based* when it is constructed from a WSPD but other transformations may be subsequently applied. For instance, nobody knows how to get a WSPD-spanner with bounded degree without applying subsequent transformations to a WSPD-spanner. Arya et al. [4, 5] obtained several results regarding these properties in WSPD-based spanners. Narasimhan and Smid [22] detailed proofs of the bounds found by Arya et al. Aryal et al. showed how to construct a WSPD-based spanner with maximum degree $O(1/(t-1)^{2d-1})$. Using Callahan and Kosaraju's construction of the WSPD and by carefully choosing the representatives, they proved that there exist WSPD-spanners with a diameter of at most $2\log(n) - 1$. In this thesis, the logarithm is base 2. Furthermore, they showed that the same construction produces spanners of weights $O(w(MST)\log n)$. Using *dumbbell trees*, which are an extension of Callahan and Kosaraju's construction of the WSPD, they also showed constructions of spanners with constant bounded diameter and sub-quadratic number of edges. See Table 1 for detailed bounds. Finally, using these results on spanners with constant bounded diameter, they showed how to construct spanners with bounded degree, sub-quadratic number of edge, $O(\log n)$ diameter, and weight of $O(w(MST)\log^2 n)$. It is unknown whether the weight can be decreased to $O(w(MST)\log n)$.

After their discovery of the well-separated pair decomposition, Callahan and Kosaraju [12] explored approximate and exact *Euclidean minimum spanning trees (EMST)*. An *ϵ-approximation of the EMST* of a point set is a spanning tree of the point set that has a weight of at most $(1+\epsilon)w(EMST)$. They gave a simple ϵ-approximation which consists of computing the WSPD-spanner and then computing the mininum spanning tree of that spanner. The value of s is set to $4/\epsilon$. This approximation is computable in time $O(\epsilon^{-d} n \log n)$ which is better than generic MST algorithms in $d > 2$ dimensions computing MSTs in $O(n^2 \log n)$ time. They also gave a more complex ϵ-approximation independent of s computable in

Diameter	Number of edges
2	$O\left(\frac{\log(1/(t-1))}{(t-1)^d} n \log n\right)$
3	$O\left(\frac{\log(1/(t-1))}{(t-1)^d} n \log\log n\right)$
$2k$ for $k \geq 4$	$O\left(\frac{\log(1/(t-1))}{(t-1)^d} 2^k n \alpha_k(n)\right)$
$O(\alpha(n))$	$O\left(\frac{\log(1/(t-1))}{(t-1)^d} n\right)$
At most $2\log(n) - 1$	$O(s^d n)$

where $\alpha : \mathbb{N} \to \mathbb{N}$ is the inverse Ackermann function and $\alpha_k : \mathbb{N} \to \mathbb{N}$ is defined as follows.

$$\begin{aligned} \alpha_{2k}(n) &= \min\{s \geq 0 : A_k(s) \geq n\} \\ \alpha_{2k+1}(n) &= \min\{s \geq 0 : B_k(s) \geq n\} \end{aligned}$$

$$\begin{aligned} A_0(n) &= 2n, \text{ for all } n \geq 0, \\ A_k(n) &= \begin{cases} 1 & \text{if } k \geq 1 \text{ and } n = 0, \\ A_{k-1}(A_k(n-1)) & \text{if } k \geq 1 \text{ and } n \geq 1. \end{cases} \\ B_0(n) &= n^2, \text{ for all } n \geq 0, \\ B_k(n) &= \begin{cases} 2 & \text{if } k \geq 1 \text{ and } n = 0, \\ B_{k-1}(B_k(n-1)) & \text{if } k \geq 1 \text{ and } n \geq 1. \end{cases} \end{aligned}$$

Table 1: List of WSPD-based spanners with bounded diameters

$O(n \log n + (\epsilon^{-d/2} \log \frac{1}{\epsilon})n)$ time. Agarwal et al. [1] presented a relation between the EMST and the *bichromatic closest pair*. In the bichromatic closest pair problem, we have n blue points and m red points and the goal is to find the closest pair containing a blue point and a red point. Let $T_d(n, m)$ be the time to compute the bichromatic closest pair. Callahan and Kosaraju [12] showed how, using the WSPD, to compute the EMST of a point in $O(T_d(n, n) \log n)$ time in general and in $O(T_d(n, n))$ if $T_d(n, n) = \Omega(n^{1+\alpha})$ for a constant $\alpha > 0$. This is not better than the algorithm found by Agarwal et al. but gives some insight to the relation of the EMST problem and the bichromatic closest pair problem.

For a complete review on WSPDs with all the theorems and proofs, see the book by Narasimhan and Smid [22].

Chapter 3

Results

3.1 Preliminaries – Construction of the WSPD

For the rest of this thesis, our setting is the Euclidean plane i.e. $\mathbb{R}^2$. There are many ways to construct a WSPD (for instance, using quadtrees [18]). In our setting, we use the construction by Callahan and Kosaraju [14] which is based on a data structure called the *split tree*. To define the concept of a split tree, we first need to define the concept of *bounding box*. A bounding box of a point set S, denoted $R(S)$, is the smallest axis-parallel rectangle containing S.

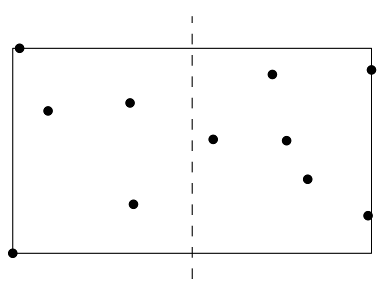
The split tree is a binary tree defined as follows. Take the bounding box $R(S)$ of the point set S and store it at the root u of the split tree. Then, split $R(S)$ on its longest side and store the bounding boxes of the two resulting subsets of S in the children of u. Repeat this recursively for each child until the leaves are the points of S. The set of points in the subtree rooted at node u is denoted S_u. We also use the notation R_u interchangeably with $R(S_u)$ to talk about the bounding box of S_u. To summarize, each internal node u of the split tree stores its bounding box R_u, and pointers to its two children. Each leaf stores a point of S which will be considered as the bounding box of u. See Algorithm 1 for the construction of a split tree.

A WSPD of a point set S is then computed using the split tree of S. Let T be the split tree of S and $s > 0$ be the desired separation ratio. Let v and w be two nodes of T. We compute whether S_v and S_w are well-separated with respect to s by using the bounding boxes R_v and R_w instead of S_v and S_w. Then, a WSPD for S is computed by calling ComputeWSPD(T, s) (refer to Algorithm 2), which

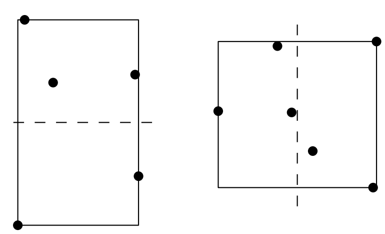
Algorithm 1 SplitTree(S)

Input: A point set S.
Output: The (root of the) split tree of S.

Let u be an empty node.
if $|S| = 1$ **then**
  Store the only point of S in u. // Note that we consider this point to be the bounding box R_u
else
  Compute the bounding box $R(S)$
  Split $R(S)$ along its longest side into two same-size rectangles R_1 and R_2.
  $S_v := S \cap R_1$
  $S_w := S \setminus S_v$
  $v :=$ SplitTree(S_v)
  $w :=$ SplitTree(S_w)
  Store v and w as the left and right children of u, respectively.
  $R_u := R(S)$
end if
return u



(a) First split



(b) Following splits

Figure 1: Illustration of the split tree

Algorithm 2 ComputeWSPD(T, s)

Input: The split tree T, and the separation ratio s.
Output: A WSPD.

for each internal node u of the split tree T **do**
  Let v and w be the left and right children of u, respectively.
  FindPairs(v, w, s)
end for

calls FindPairs(v, w) (refer to Algorithm 3). Callahan and Kosaraju proved that

Algorithm 3 FindPairs(v, w, s)

Input: Two nodes v and w of a split tree, and the separation ratio s.
Output: A set of well-separated pairs $\{\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}\}$ such that for any point $p \in S_v$ and any point $q \in S_w$, there is a unique pair $\{A_i, B_i\}$, $1 \leq i \leq m$, such that $p \in A_i$ and $q \in B_i$.

```
if S_v and S_w are well-separated with respect to s then
    Report the pair {S_v, S_w}
else if L_max(R_v) ≤ L_max(R_w) then // Let the function L_max(·) be the longest side of a bounding box.
    Let w_l and w_r be the left and right children of w, respectively.
    FindPairs(v, w_l)
    FindPairs(v, w_r)
else
    Let v_l and v_r be the left and right children of v, respectively.
    FindPairs(v_l, w)
    FindPairs(v_r, w)
end if
```

this algorithm produces a linear number of pairs [14]. The following lemma gives properties about the points in a pair of a WSPD.

Lemma 1 (Callahan and Kosaraju [12]). *Let $\{A, B\}$ be a well-separated pair with respect to the separation ratio $s > 0$. Let $p, p', p'' \in A$ and $q, q' \in B$. Then,*

- $|p'p''| \leq (2/s)|pq|$
- $|p'q'| \leq (1 + 4/s)|pq|$

3.2 Construction of t-Spanners Using WSPDs

In this section, we show how to construct a WSPD-spanner on which our routing results are based. We also prove some useful geometric lemmas concerning these spanners. Callahan and Kosaraju's [14] classical construction of a spanner given a WSPD proceeds as follows: for each well-separated pair $\{A, B\}$, select an arbitrary point $a \in A$ as a *representative* of the set A and an arbitrary point $b \in B$ as a representative of the set B and add the edge ab to the graph. Callahan and Kosaraju

[12] proved that any WSPD-spanner constructed this way has a spanning ratio of at most $1 + 8/(s-4)$, where s is the separation ratio of the WSPD.

To facilitate the design of our routing algorithm, rather than selecting an arbitrary point as the representative of a set in a pair, we choose the rightmost point as the representative. If there is more than one rightmost point, we choose the topmost point among the rightmost ones. The following definitions define three types of spanners based on WSPDs, depending on how the WSPD was constructed and how the representatives are chosen.

Definition 2 (AW-Spanner)**.** *An AW-Spanner (AW for "**A**rbitrary **W**SPD") is a spanner based on a WSPD where the choice of the representative of each set in a well-separated pair of the WSPD is arbitrary.*

Definition 3 (ASW-Spanner)**.** *An ASW-Spanner (ASW for "**A**rbitrary representative, **S**plit tree, **W**SPD") is a spanner based on a WSPD computed with a split tree where the choice of the representative of each set in a well-separated pair of the WSPD is arbitrary.*

Definition 4 (RSW-Spanner)**.** *An RSW-Spanner (RSW for "**R**ightmost representative, **S**plit tree, **W**SPD") is a spanner based on a WSPD computed with a split tree. Moreover, the representative of each set in a well-separated pair of the WSPD is chosen such that it is the rightmost point of the set. If there is more than one rightmost point, the topmost point among the rightmost ones is chosen.*

In this thesis, we explain how to do local routing in RSW-Spanners. One reason for using the pairs of a WSPD constructed with a split tree is the fact that the number of pairs is linear in the number of points in S, if we assume that s is a constant [14]. This implies that the resulting spanner has a linear number of edges. Moreover, the way that representatives are chosen in a RSW-Spanner gives us several geometric properties that can be exploited. Thus, in the remainder of the chapter, unless stated otherwise, we focus on RSW-Spanners.

In Theorem 1, by exploiting properties of the split tree, we prove that the spanning ratio of ASW-Spanners is at most $1 + 4/(s-4) + 4/s$ which is a slight improvement over the spanning ratio of $1 + 8/(s-4)$, shown for AW-Spanners. In Theorem 2, we make a further improvement to $1+4/(s-2)+4/s$ for RSW-Spanners. Before proving this, we begin with some helper lemmas.

Lemma 2. *Let u and v be any two nodes in a split tree. If u is an ancestor of v, then $S_v \subset S_u$. Otherwise, $S_v \cap S_u = \emptyset$.*

Proof. Let S_u and S_v be two point sets associated to u and v. When the bounding box of a node x is split into two in the construction of the split tree (refer to Algorithm 1), the sets associated to the two children of x are disjoint and are subsets of the set associated to x. Hence, S_u and S_v are either disjoint sets or one is a subset of the other. □

Lemma 3. *In an RSW-Spanner, consider two sets A and C each from a different pair of the WSPD. Let a be a representative of A. If $C \subseteq A$ and $a \in C$, then a is also the representative of C.*

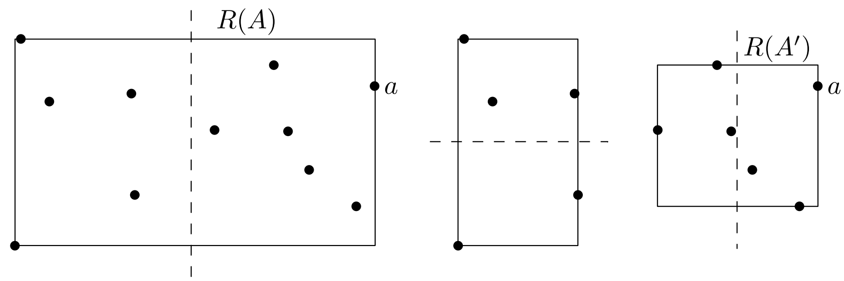
Proof. Since $C \subseteq A$ and a is the rightmost, topmost point of A, then a is also the rightmost, topmost point in C. Thus, a is the representative of C. □

Lemma 4. *In an RSW-Spanner, let A be a set in a pair from the WSPD and let $a, x \in A$ be two points such that a is the representative of A and $x \neq a$. There is a well-separated pair $\{C, D\}$ such that:*

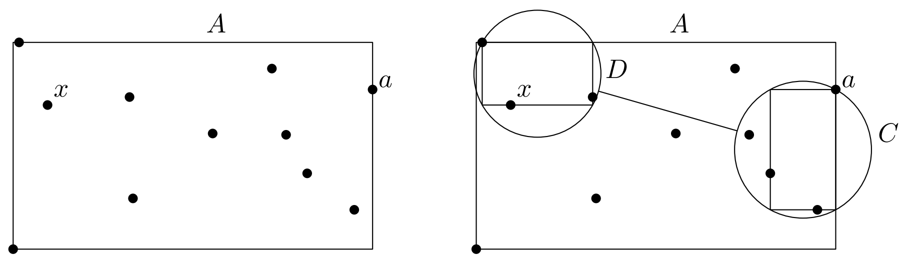
- *$a \in C$;*
- *$x \in D$;*
- *a is the representative of C;*
- *C is a proper subset of A;*
- *D is a proper subset of A.*

(refer to Figure 2b).

Proof. Let $\{C, D\}$ be the pair that separates a from x. Therefore, C and D must be disjoint. Since a and x are in A, we have that C and D are both disjoint subsets of A by Lemma 2. We have that a is the representative of C by Lemma 3. See Figure 2b for an illustration. □



(a) Illustration of Lemma 3. The split of $R(A)$ and the two bounding boxes obtained during the construction of the split tree. Notice that a is the representative of $R(A')$.



(b) Illustration of Lemma 4. The pair $\{C, D\}$ separating a from x.

Figure 2: Illustration of Lemma 3 and Lemma 4

Let $[x, a]$ be an edge in an RSW-Spanner such that a is the representative of a bounding box A containing x. Let C be a bounding box smaller than A that contains x in the WSPD. Lemma 5 states that the representative c of C has an edge to the representative a of A. Lemma 6 states that a has an edge to the representatives of some bounding boxes such that their union contains all the points of which C is well-separated from.

Lemma 5. *In an RSW-Spanner, let C be a set in a pair from the WSPD and let $c, x \in C$ be two points such that c is the representative of C and $x \neq c$. Let A be a set in a pair from the WSPD such that $C \subset A$ and a point $a \in A$ is the representative of A. If $[x, a]$ is an edge, then $[c, a]$ is an edge (Refer to Figure 3).*

Proof. We claim that during the construction of the WSPD, there is a call to FIND-PAIRS(u, v), where $R_u = R(C)$, $a \in S_v$ and $S_v \subset A$. Before proving our claim, let us show how applying the claim proves the lemma. By Lemma 3, a is the representative of all sets containing a in all pairs reported from this call since a is a representative of R_v. Similarly, the representative c of C is also the representative of all sets containing

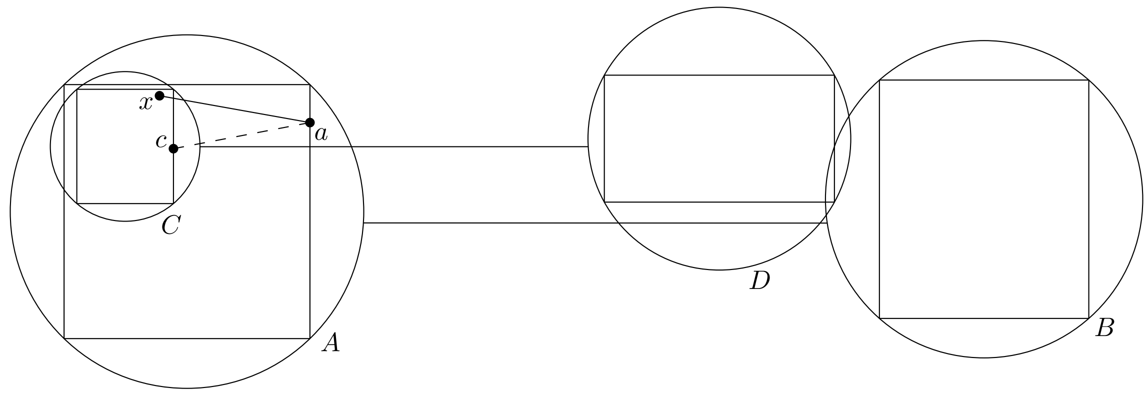


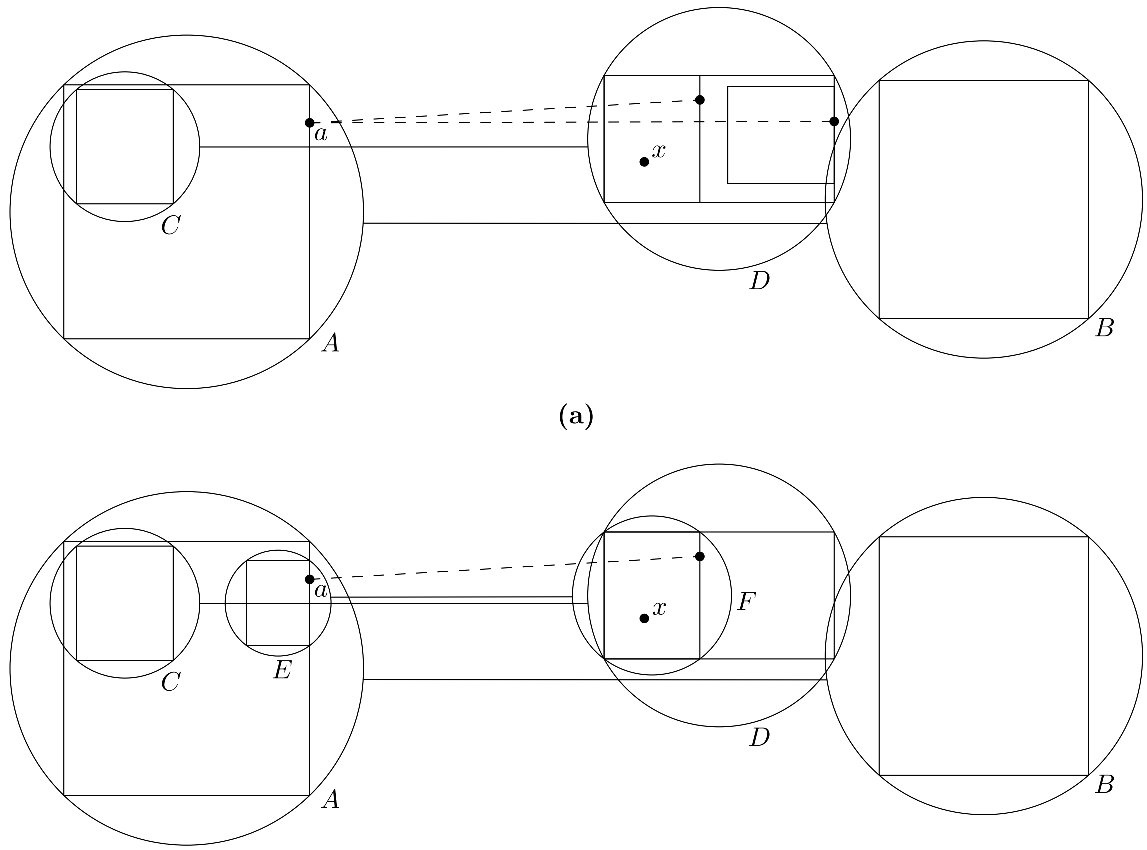
Figure 3: Illustration of Lemma 5. Connected circles represent pairs in the WSPD.

c in all pairs reported from this call. Thus, the representative c of C has an edge to a.

Let us now prove our claim. Let FINDPAIRS(u', v') be the call where the pair separating x from a is reported such that $x \in S_{u'}$ and $a \in S_{v'}$ without loss of generality. Since x is not the representative of $R(C)$, x is not the rightmost, topmost point of $R(C)$. Therefore, $R(C)$ must be the bounding box of an ancestor of u', and $S_{u'} \subset C$. Thus, since $S_{u'} \subset C$, and $a \notin C$, there must have been a call to FINDPAIRS(u, v) where $R_u = R(C)$, and $a \in S_v$. By Lemma 2, since $C \subset A$, $a \in A$, and two sets in a pair are disjoint, we get $S_v \subset A$. □

Lemma 6. *In an RSW-Spanner, let $\{A, B\}$ and $\{C, D\}$ be two distinct pairs from the WSPD, such that $C \subset A$. Let a be the representative of A. Let x be any point in D and let $\{E, F\}$ be the unique pair from the WSPD separating $a \in E$ from $x \in F$. Then, a is the representative of E (Refer to Figure 4).*

Proof. We consider two cases. Either $x \in A$ or $x \notin A$. If $x \in A$, the result follows from Lemma 4. Otherwise, if $x \notin A$, consider the call to FINDPAIRS(u', v') that reports the pair $\{C, D\}$. From the algorithm FINDPAIRS, we know that $R_{u'} = R(C)$ and $R_{v'} = R(D)$. By Lemma 2, since $x \notin A$, we know that $D \cap A = \emptyset$. Since $C \subset A$, $D \cap A = \emptyset$ and $R_{v'} = R(D)$, there must have been a call to FINDPAIRS(u, v) that led to the call FINDPAIRS(u', v'), where u is an ancestor of u', $R_u = R(A)$ and $D \subset S_v$. By Lemma 3, since a is the representative of A, we get that a is the representative of



(a)

(b)

Figure 4: Illustration of Lemma 6

all sets containing a in all pairs reported from the call to FindPairs(u, v). Thus, a is representative of the set separating a from $x \in D$. $\square$

Algorithm 4 finds a path between p and q in an AW-Spanner and is derived from the proof of Theorem 9.2.1 by Narasimhan and Smid in [22].

Algorithm 4 FindPath(p, q)

Precondition: $p \neq q$

Let $\{A, B\}$ be the unique pair in the WSPD separating $p \in A$ from $q \in B$.
Let a and b be the representatives of A and B.
return FindPathRec(p, a, A), FindPathRec(b, q, B)

We consider the level of recursion to be 1 during the execution of the first call of FindPathRec and k when executing the k-th call in the execution stack of FindPathRec from an initial call of FindPath.

Algorithm 5 FINDPATHREC(v, w, E)

Precondition: $v, w \in E$,
either v or w is the representative of E.
if $v = w$ **then**
return v
else
Let $\{C, D\}$ be the pair in the WSPD separating $v \in C$ from $w \in D$.
Let c and d be the representatives of C and D, respectively.
return FINDPATHREC(v, c, C), FINDPATHREC(d, w, D)
end if

Lemma 7. *Let $p, q \in S$. Consider a call to* FINDPATH*(p, q) in an ASW-Spanner of S. Consider the call to* FINDPATHREC*(v, w, E) at recursion depth $k \geq 1$. For any two points $e, f \in E$, $|ef| \leq (2/s)^k |pq|$.*

Proof. We prove this lemma by induction.

Base case: $k = 1$

Let $\{A, B\}$ be the pair that separates $p \in A$ from $q \in B$. Since $k = 1$, $E = A$ or $E = B$, e and f are either both in A or both in B. By Lemma 1, we get that $|ef| \leq (2/s)|pq|$.

Induction step: Let $k > 1$. Let FINDPATHREC(v', w', E') be the parent call of FINDPATHREC(v, w, E). Thus, the call FINDPATHREC(v', w', E') is at level $k - 1$. Consider two arbitrary points $e', f' \in E'$. By the induction hypothesis, we have $|e'f'| \leq (2/s)^{k-1}|pq|$.

Let $\{C', D'\}$ be the pair in the WSPD separating $v' \in C'$ from $w' \in D'$. By definition of FINDPATHREC, $E = C'$ or $E = D'$. Thus, e and f are either both in C' or both in D'. Observe that $v', w' \in E'$ according to the preconditions of FINDPATHREC(v', w', E'). By Lemma 1, we get that $|ef| \leq (2/s)|v'w'|$. From the induction hypothesis, we have that $|v'w'| \leq (2/s)^{k-1}|pq|$. Thus, we get that $|ef| \leq (2/s)(2/s)^{k-1}|pq| = (2/s)^k|pq|$. □

The following two theorems give upper bounds on the spanning ratios of ASW-Spanners (Theorem 1) and RSW-Spanners (Theorem 2). The only difference is in the choice of representatives. The proofs of Theorems 1 and 2 are similar.

Theorem 1. *The spanning ratio t of an ASW-Spanner is at most $4/(s-4)+4/s+1$.*

Proof. We find an upper bound on the spanning ratio of a path from p to q by analyzing the path found by FINDPATH(p, q). Consider only one of the two calls to FINDPATHREC in FINDPATH(p, q). Let FINDPATHREC(v, w, E) be the considered call. Notice that we are considering all subsequent calls to FINDPATHREC following the call to FINDPATHREC(v, w, E) in FINDPATH(p, q). Since we are only considering one call to FINDPATHREC in FINDPATH(p, q), each level $k \geq 1$ of recursion in FINDPATHREC(v, w, E) can have at most 2^{k-1} instances, i.e. there are at most 2^{k-1} edges $[c, d]$ at depth k in the recursion. Notice that $v, w \in E$ according to the preconditions of FINDPATHREC(v, w, E). Therefore, by Lemma 2, c and d are also both in E. From Lemma 7, we get $|cd| \leq (2/s)^k |pq|$. Thus, the sum of the length of all edges $[c, d]$ at level k is bounded by $2^{k-1}\left(\frac{2}{s}\right)^k |pq|$. Then, if we sum up the lengths of all edges $[c, d]$ from level 1 to a maximum depth m, we get

$$\sum_{i=1}^{m} 2^{i-1} \left(\frac{2}{s}\right)^i |pq| \leq \sum_{i=1}^{\infty} 2^{i-1} \left(\frac{2}{s}\right)^i |pq| = \frac{2}{s-4}|pq|.$$

Let $\{A, B\}$ be the pair separating $p \in A$ and $q \in B$. Let $a \in A$ and $b \in B$ be the representatives of A and B, respectively. From Lemma 1, we have that $|ab| \leq (1 + 4/s)|pq|$.

To bound the path found by FINDPATH(p, q), we take the length of the path found by the call to FINDPATHREC(p, a, A), add the length of the edge $[a, b]$, and add the length of the path found by the call to FINDPATHREC(q, b, B). Thus, the path found by FINDPATH(p, q) has a length of at most

$$2 \cdot \frac{2}{s-4}|pq| + \left(1 + \frac{4}{s}\right)|pq| = \left(\frac{4}{s-4} + \frac{4}{s} + 1\right)|pq|.$$

□

The following theorem is similar to Theorem 1. Essentially, using Lemma 4, we show that each level k has only one edge instead of 2^{k-1} in the previous proof. Thus, we calculate the spanning ratio according to the choice of representatives of RSW-Spanners. In the following sections, we will take the path found by FINDPATH on RSW-Spanners to prove the correctness and find the routing ratio of our local routing

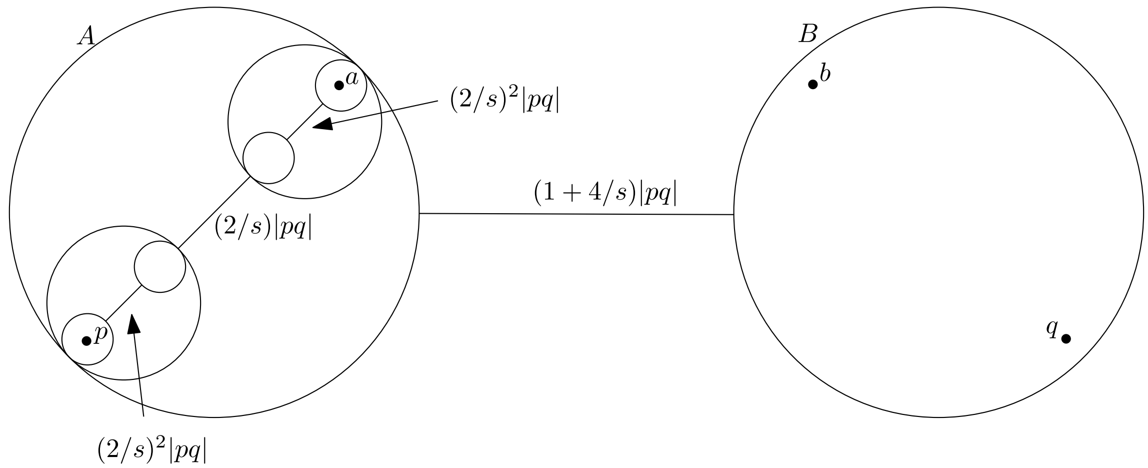


Figure 5: Illustration of Theorem 1

algorithm.

Theorem 2. *The spanning ratio t of an RSW-Spanner is at most $4/(s-2)+4/s+1$.*

Proof. We find an upper bound on the spanning ratio of a path from p to q by analyzing the path found by FINDPATH(p, q). Consider only one of the two calls to FINDPATHREC in FINDPATH(p, q). Let FINDPATHREC(v, w, E) be the considered call. Since either v or w is the representative of E, by Lemma 4, we know that either v or w is the representative of C or D and, thus, either $v = c$ or $w = d$. This means that for each level $k \geq 1$, the call to FINDPATHREC(w, d, D) returns immediately. In other words, for all $k \geq 1$, there is exactly one edge of level k. Notice that $v, w \in E$ according to the preconditions of FINDPATHREC(v, w, E). Therefore, by Lemma 2, c and d are also both in E. From Lemma 7, we get $|cd| \leq (2/s)^k|pq|$. The fact that there is exactly one edge of level k allows us to get only $(2/s)^k$ as the sum of the length of all edges at level k. This contrasts with Theorem 1 where this sum is $2^{k-1}(2/s)^k$. Then, if we sum up the length of all edges $[c, d]$ from level 1 to a maximum depth m, we find

$$\sum_{i=1}^{m} \left(\frac{2}{s}\right)^i |pq| \leq \sum_{i=1}^{\infty} \left(\frac{2}{s}\right)^i |pq| = \frac{2}{s-2}|pq|.$$

Let $\{A, B\}$ be the pair separating $p \in A$ and $q \in B$. Let $a \in A$ and $b \in B$ be the representatives of A and B, respectively. From Lemma 1, we have that $|ab| \leq (1 + 4/s)|pq|$.

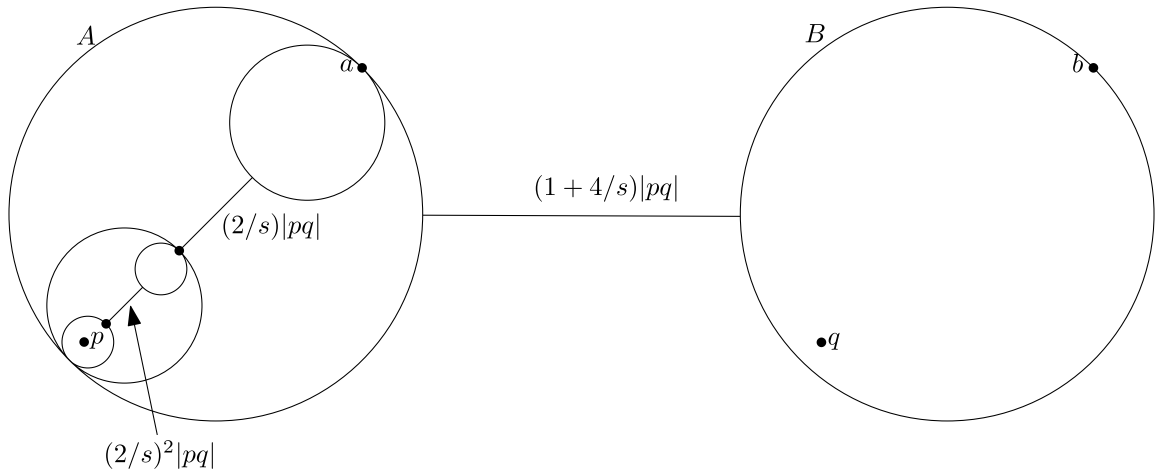


Figure 6: Illustration of Theorem 2

To bound the path found by FINDPATH(p, q), we take the length of the path found by the call to FINDPATHREC(p, a, A), add the length of the edge $[a, b]$, and add the length of the path found by the call to FINDPATHREC(q, b, B). Thus, the path found in FINDPATH(p, q) has a length of at most

$$2 \cdot \frac{2}{s-2}|pq| + \left(1 + \frac{4}{s}\right)|pq| = \left(\frac{4}{s-2} + \frac{4}{s} + 1\right)|pq|.$$

□

We now define some notation that will be useful throughout the rest of this thesis. Each pair of sets in a WSPD is associated with a pair of bounding boxes. Let $\{X, Y\}$ be the unique pair in the WSPD that separates a point $x \in X$ from a point $y \in Y$. There are two bounding boxes defined with respect to X and Y, namely $R(X)$ and $R(Y)$. To refer to these bounding boxes from the perspective of x and y, we use the following notation: $R(X)$ is referred by $B_{xy}(x)$, and $R(Y)$ is referred by $B_{xy}(y)$. Notice that $B_{xy}(x) = B_{yx}(x) = R(X)$, $B_{xy}(y) = B_{yx}(y) = R(Y)$ and $B_{xy}(y) \neq B_{xy}(x)$. Let y^* be the representative of Y. We can say that y^* is the representative of $B_{xy^*}(y^*) = B_{xy}(y) = R(Y)$. Therefore, y^* has an edge to the representative of $B_{xy^*}(y^*)$. See Figure 7 for illustration.

Furthermore, we denote by $P_t(p, q)$ the path from a point p to a point q with spanning ratio t, found by the FINDPATH algorithm in an RSW-Spanner. Let v be

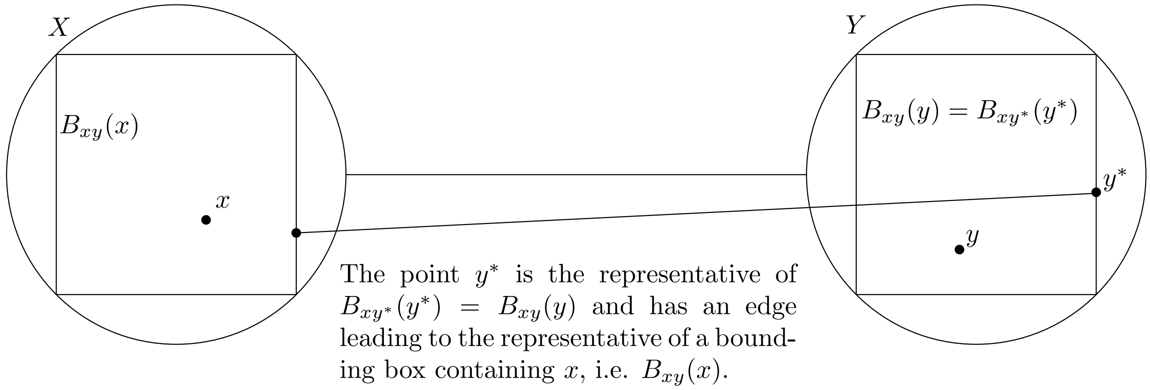


Figure 7: Illustration of the notation

a point of $P_t(p, q)$ in $B_{pq}(p)$ but not representative of $B_{pq}(p)$. Let B_v be the largest bounding box with v as representative. Suppose that p is in B_v. Lemma 8 establishes a relation between B_v and the bounding boxes containing p of the points of $P_t(p, q)$.

Lemma 8. *Consider any RSW-Spanner. Let p, q and v be three points such that:*

- *v is inside $B_{pq}(p)$;*
- *v is not the representative of $B_{pq}(p)$;*
- *p is in B_v (the largest bounding box that v is representative of).*

There must exist an edge $[d, e]$ of $P_t(p, q)$ such that $B_{de}(d)$ is the smallest bounding box containing p that is larger than B_v. Then, there is an edge between v and d.

Proof. We first argue that the edge $[d, e]$ is well-defined. Since v is inside $B_{pq}(p)$ but is not the representative of $B_{pq}(p)$, we know that B_v is smaller than and inside $B_{pq}(p)$ by Lemma 2 and 3, respectively. This implies that the set of edges $[\alpha, \beta]$ from $P_t(p, q)$ such that p is in $B_{\alpha\beta}(\alpha)$ and B_v is smaller than $B_{\alpha\beta}(\alpha)$ is non-empty. Indeed, the edge $[a, b]$ from $B_{pq}(p)$ to $B_{pq}(q)$ is in this set since $B_{ab}(a) = B_{pq}(p)$. Therefore, the edge $[d, e]$ is well-defined.

Let c be the point before d in $P_t(p, q)$. Since d is in $P_t(p, q)$, then d is representative of $B_{de}(d)$. Therefore, by Lemma 4, we know that d is the representative of $B_{pd}(d)$ and c is the representative of $B_{pd}(p)$ since it is the unique pair separating p from d. Then, c is the representative of $B_{cd}(c) = B_{pd}(p)$ such that p is in $B_{cd}(c)$. Refer to Figure 8 for illustration.

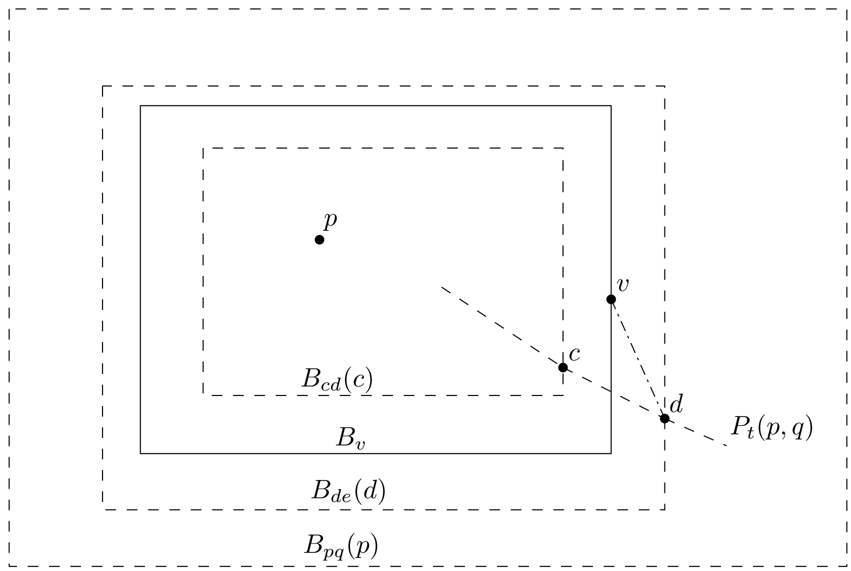


Figure 8: Illustration for the proof of Lemma 8. The dashed line segments represent edges between points in $P_t(p, q)$. The boxes with dashed borders represent boxes of $P_t(p, q)$. The dash dotted line segment represents the edge between v and d.

Because $B_{de}(d)$ is the smallest bounding box containing p larger than B_v, $c \in B_{cd}(c) \subseteq B_v$. If $v = c$ is the representative of $B_{cd}(c)$, then v has an edge to d. Otherwise, $B_{cd}(c) \subset B_v$ and we apply Lemma 5 in the following way. We have that c is in B_v, $B_v \subset B_{de}(d)$, d is the representative of $B_{de}(d)$ and there is an edge between c and d. Therefore, there is an edge from v to d by Lemma 5. $\square$

The next theorem presents a non-trivial lower bound on the spanning ratio of an RSW-Spanner.

Theorem 3. *For any $s > 0$, there exist an RSW-Spanner with a spanning ratio arbitrarily close to $1 + 8/s$.*

Proof. Let $0 < \epsilon < \pi$ be a real number. Let $S = \{p, p', q, q'\}$ be a point set such that:

$$\begin{aligned}
p &= (\cos(\pi/2 + \epsilon), \sin(\pi/2 + \epsilon)), \\
p' &= (\cos(-\pi/2 + \epsilon), \sin(-\pi/2 + \epsilon)), \\
q &= (\cos(-\pi/2 - \epsilon), \sin(-\pi/2 - \epsilon) + s + 2), \\
q' &= (\cos(\pi/2 - \epsilon), \sin(\pi/2 - \epsilon) + s + 2)
\end{aligned}$$

(refer to Figure 9).

Let $A = \{p, p'\}$ and $B = \{q, q'\}$. By construction, there is a pair $\{A, B\}$ in the WSPD. Again by construction, p' is the representative of $R(A)$ and q' is the representative of $R(B)$. Hence, the only path between p and q is $pp'q'q$. We have

$$\begin{aligned}\lim_{\epsilon\to 0} |pp'| &= \lim_{\epsilon\to 0} \sqrt{(\cos(\pi/2+\epsilon) - \cos(-\pi/2+\epsilon))^2 + (\sin(\pi/2+\epsilon) - \sin(-\pi/2+\epsilon))^2} \\ &= 2.\end{aligned}$$

Similarly,

$$\begin{aligned}\lim_{\epsilon\to 0} |qq'| &= 2, \\ \lim_{\epsilon\to 0} |p'q'| &= s+4, \\ \lim_{\epsilon\to 0} |pq| &= s.\end{aligned}$$

Thus, the spanning ratio of the path between p and q and, therefore, the spanning ratio of the graph approaches

$$\lim_{\epsilon\to 0} \frac{|pp'| + |p'q'| + |q'q|}{|pq|} = \frac{2+s+4+2}{s} = \frac{s+8}{s} = 1 + \frac{8}{s}$$

as ϵ approaches 0. □

3.3 2-Local Routing Algorithm

3.3.1 The Algorithm

Recall that we defined local routing as a function $f(v, p, q, \mathcal{N}(v)) = w$ that takes the current point v on the path and decides the next point w on the path using information about the source p, the destination q and the neighbors $\mathcal{N}(v)$ of v. In our setting, we allow additional information to be stored. In this section, we define this additional information and then define our algorithm. Essentially, a local routing algorithm finds a path from p to q by making choices using only local information

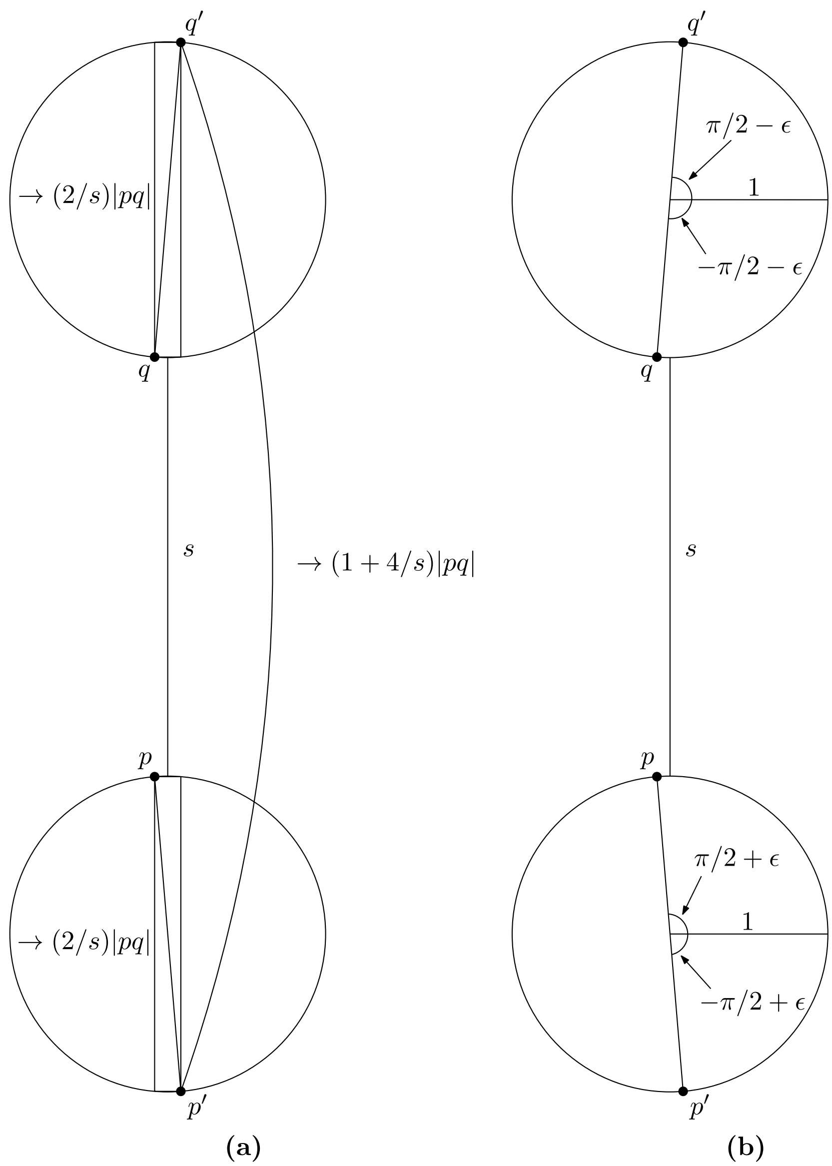


Figure 9: Illustration of Theorem 3 (a) with the bounding boxes with the lengths of the edges and (b) with the angles.

available at each point of the graph. The goal of this thesis is to do local routing in spanners constructed using WSPDs. To that end, we chose to work on RSW-Spanners since they satisfy useful geometric properties we can exploit. We now describe the additional information that is available at each point, then we describe our local routing algorithm.

Let v be the current point of the routing path. For all neighbors d of v, and for all neighbors e of d, we suppose that the following information is available at v:

- the edge $[v, d]$ together with $B_{vd}(v)$ and $B_{vd}(d)$;
- the edge $[d, e]$ together with $B_{de}(d)$ and $B_{de}(e)$.

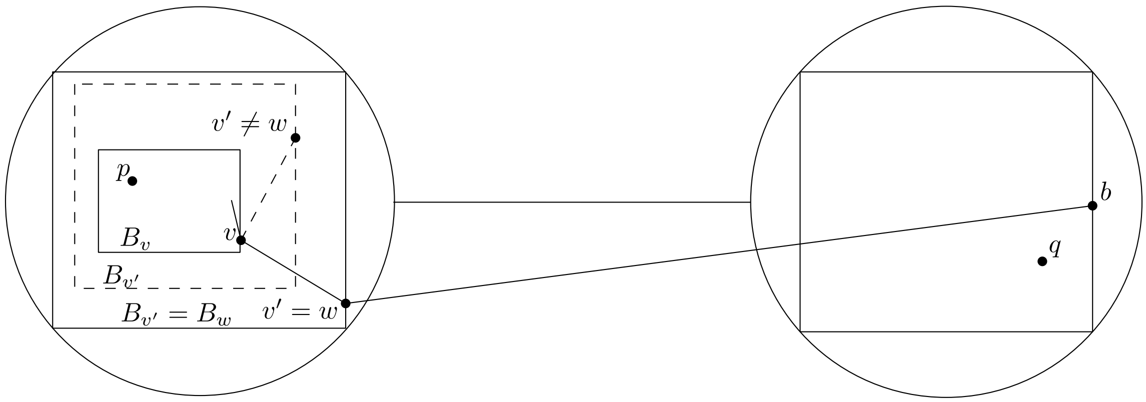
Notice that the algorithm knows $B_{de}(d)$ and $B_{de}(e)$ even though the current point is v. The fact that we know $B_{de}(e)$ makes our algorithm 2-local since e is 2 hops away from v. In Section 3.4, we will modify our algorithm so that it does not need to know $B_{de}(e)$. This will lead to a 1-local routing algorithm with a slightly larger routing ratio.

We want to find a path between two points $p \in S$ and $q \in S$. Let $\{A, B\}$ be the unique pair in the WSPD separating p from q. Let a and b be the representatives of A and B, respectively. The goal for our algorithm is to find a path from p to a, take the edge $[a, b]$, and then, find a path b from q such that the length of the path from p to q is at most $t|pq|$, where $t > 1$ is a spanning ratio.

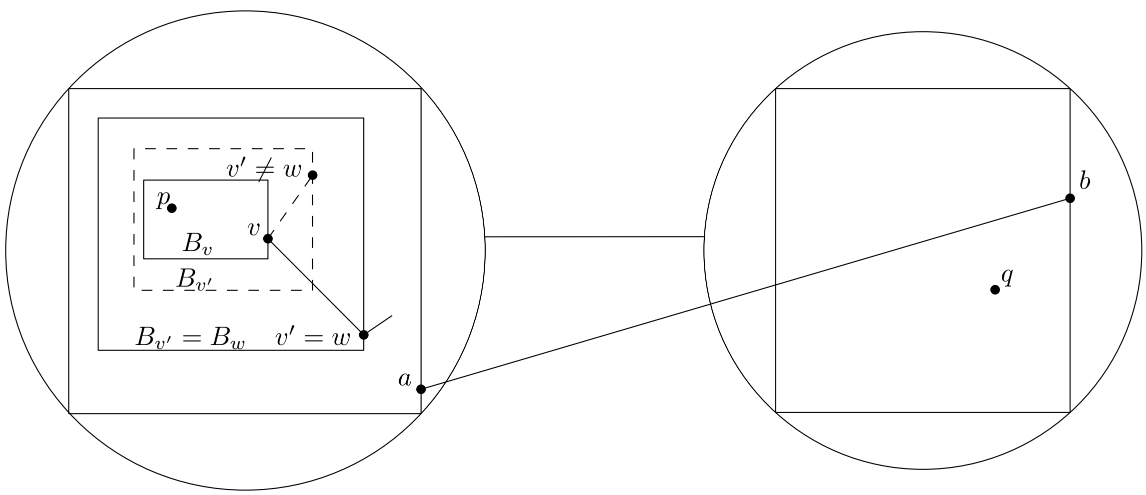
To find a path from p to a (what we call the Enlarging step), we use the following strategy. Let v be the current point on the path from p to a produced by our algorithm (at the beginning $v = p$). Here is how our algorithm selects the next edge. The algorithm verifies if v has a neighbor w such that w is the representative of $B_{pq}(p)$. If such a w exists, then the edge $[v, w]$ is chosen by the algorithm. See Figure 10a for illustration. Otherwise, consider the follow set:

$$\mathcal{V} = \{v' \in \mathcal{N}(v) \mid p \in B_{v'},\ v' \text{ is not the representative of } B_{v'q}(v')\}$$

where $\mathcal{N}(v)$ denotes the set of neighbours of v. In the proof of Lemma 11, we prove that $\mathcal{V}$ is non-empty and that for any $v' \in \mathcal{V}$, $B_{v'}$ is contained in $B_{pq}(p)$. Then,



(a) Illustration of the Enlarging step of Algorithm 6, where w is the representative of $B_{pq}(p)$



(b) Illustration of the Enlarging step of Algorithm 6, where w has no edge leading to the representative of a bounding box containing q.

Figure 10: Illustration of the Enlarging step of Algorithm 6.

the next edge chosen by our algorithm is the edge $[v, w]$ such that the size of B_w is maximized among all $w \in \mathcal{V}$. See Figure 10b for illustration.

Upon reaching a, we take the edge $[a, b]$. To find a path from b to q (what we call the Reducing step), notice that b must the representative of $B_{bq}(b)$. Let w be the representative of $B_{bq}(q)$. The algorithm takes the edge $[b, w]$. Then, we repeat this procedure until the algorithms arrives at q.

Our algorithm is summarized in Algorithm 6. Note that sizeof$(B_{v'})$ denotes the area of $B_{v'}$.

Lemma 9. *Algorithm 6 is 2-local.*

Algorithm 6 FINDPATHTWOLOCAL(v, p, q)

Input: the current point v,
the source p,
the destination q.

Output: The next point w on the path.

if there is an edge $[v, v']$ where v' is the representative of $B_{vq}(q)$ **then** // Reducing step
  $w \leftarrow v'$
else// Enlarging step
  if there is an edge $[v, v']$ where v' is the representative of $B_{pq}(p)$ **then**
    $w \leftarrow v'$
  else
    $\forall v' \in \mathcal{N}(v)$, let $B_{v'}$ be the largest bounding box that v' is the representative of.
    Let $\mathcal{V} = \{v' \in \mathcal{N}(v) \mid p \in B_{v'},\ v'$ is not the representative of $B_{v'q}(v')\}$
    $w \leftarrow \operatorname{argmax}_{v' \in \mathcal{V}} \operatorname{sizeof}(B_{v'})$
  end if
end if
return w

Proof. In Algorithm 6, the information used is the location of the neighbors of v, the bounding boxes of: the current point v, every neighbor v' of v, and every neighbor v'' of every neighbor of v. Notice that the Algorithm 6 needs to know v'' in order to test whether v' is the representative of $B_{pq}(p)$ or $B_{v'q}(v')$. The knowledge of v'' makes the algorithm 2-local. □

Lemma 10. *In Algorithm 6, the total amount of information stored in the vertices is equal to $O(s^2n^2B)$, where B is the maximum number of bits to store a bounding box.*

Proof. Let v be the current point of the routing path. For all neighbors d of v, and for all neighbors e of d, the following information is available at v:

- the edge $[v, d]$ together with $B_{vd}(v)$ and $B_{vd}(d)$;
- the edge $[d, e]$ together with $B_{de}(d)$ and $B_{de}(e)$.

Since there is a constant number of bounding boxes stored for each edge, we just need to count the number of edges stored at v and multiply it by the maximum number of bits B to store a bounding box.

Callahan and Kosaraju [14] proved that the number of pairs in a WSPD computed by the algorithm ComputeWSPD is $O(s^2n)$. Thus, there is $O(s^2n)$ edges in the graph. Consider the directed versions of these edges i.e. $[x, x']$ is a different edge than $[x', x]$. There is still a linear number of directed edges in the graph. The destination of each directed edge can have at most a linear number of edges. Thus, the total size of the local information stored in all vertices is $O(s^2n)O(n)B = O(s^2n^2B)$ bits. □

Observe that a bounding box is uniquely defined by at most four points. Thus, in the statement of Lemma 10, B is at most the number of bits required to store four points.

3.3.2 Correctness

In this section, we prove the correctness of Algorithm 6 (refer to Theorem 4). For the rest of this thesis, we denote by $P_t(p, q)$ the path from p to q with spanning ratio t, found by the FindPath algorithm, and, we denote by $P_6(p, q)$ the path from p to q found by Algorithm 6.

The following lemma is used to prove the correctness of Algorithm 6 and to establish an upper bound on the routing ratio of Algorithm 6 (refer to Theorem 5).

Lemma 11. *Algorithm 6 finds a path in an RSW-Spanner from p to the representative a of $B_{pq}(p)$ by repeatedly applying the Enlarging step (Lines 4 to 9 of Algorithm 6).*

Proof. Let v be the current point. If v has an edge leading to the representative a of $B_{pq}(p)$, then Line 5 of the Enlarging step of Algorithm 6 choose the edge $[v, a]$.

Otherwise, we prove that each edge $[v, w]$ taken in Line 9 of the Enlarging step of Algorithm 6 leads to the representative of a bounding box B_w that contains p and is larger than B_v but not larger than $B_{pq}(p)$. Thus, Algorithm 6 finds a path from p to the representative of $B_{pq}(p)$. Recall that in Algorithm 6, we define

$$\mathcal{V} = \{v' \in \mathcal{N}(v) \mid p \in B_{v'},\, v' \text{ is not the representative of } B_{v'q}(v')\}.$$

Suppose that the current point v is inside but is not the representative of $B_{pq}(p)$. From Lemma 8, we get that v has an edge to the representative of a point of $P_t(p, q)$

that has a bounding box larger than B_v. This proves that there is always a choice of edges in the Enlarging step such that B_w contains p and is larger than B_v but not larger than $B_{pq}(p)$. Since any point w inside but not representative of $B_{pq}(p)$ cannot be the representative of $B_{wq}(w)$, $\mathcal{V}$ is non-empty.

Now, we prove that the next edge $[v, w]$ is chosen such that w is inside $B_{pq}(p)$. We prove this by contradiction. Suppose Algorithm 6 takes the edge $[v, w]$ where w is outside of $B_{pq}(p)$. Therefore, w must be the representatives of $B_{pq}(p)$ or must be in $\mathcal{V}$. Since w is outside of $B_{pq}(p)$, it cannot be the representative of $B_{pq}(p)$. Thus, it must be in $\mathcal{V}$. Since p is in B_w and w is outside of $B_{pq}(p)$, we have that B_w is larger than $B_{pq}(p)$. Since the representative of $B_{pq}(p)$ has an edge to the representative of a bounding box containing q, from Lemma 6, we also get that w has an edge to the representative of a bounding box containing q which contradicts the definition of $\mathcal{V}$.

Because $v = p$ is inside $B_{pq}(p)$ in the first call of Algorithm 6, we then get that each edge $[v, w]$ taken in the Enlarging step of Algorithm 6 leads to the representative of a bounding box B_w that is larger than B_v but not larger than $B_{pq}(p)$. $\square$

Once the representative a of $B_{pq}(p)$ is found, then Algorithm 6 follows the edge to the representative b of $B_{pq}(q)$.

Lemma 12. *Algorithm 6 finds a path in an RSW-Spanner from the representative b of $B_{pq}(q)$ to q by repeatedly applying the Reducing step (Lines 1 to 2 of Algorithm 6). Moreover, the path taken from b to q is the same as the path found by the algorithm* FINDPATH.

Proof. By Lemma 4, b is the representative of $B_{bq}(b)$. Let x be the representative of $B_{bq}(q)$. Thus, the Reducing step takes the edge $[b, x]$. Furthermore, since b is the representative of $B_{bq}(b)$, the algorithm FINDPATH also takes the edge $[b, x]$. Then, both algorithms repeat this step until q is found. $\square$

The following theorem follows from Lemmas 11 and 12.

Theorem 4. *Algorithm 6 finds a path in an RSW-Spanner from p to q.*

3.3.3 Routing Ratio

In this section, we find an upper bound on the routing ratio of Algorithm 6.

Lemma 13. *Algorithm 6 finds a path in an RSW-Spanner from the representative b of $B_{pq}(q)$ to q by repeatedly applying the Reducing step. The sum of the lengths of the chosen edges is at most $\frac{2}{s-2}|pq|$.*

Proof. By Lemma 12, the Reducing step of Algorithm 6 follows exactly what the recursive algorithm FINDPATH does. In the proof of Theorem 2, we show that the length of the path between b and q is at most $\frac{2}{s-2}|pq|$. Therefore, the length of the path between b and q in the Reducing step is at most $\frac{2}{s-2}|pq|$. □

Lemma 14. *Algorithm 6 finds a path in an RSW-Spanner from p to the representative a of $B_{pq}(p)$ by repeatedly applying the Enlarging step. The sum of the lengths of the chosen edges is at most $\frac{4}{s-2}|pq|$.*

Proof. Consider the edges of $P_6(p,q)$ as directed from p to q. Thus, if $[u,v]$ is an edge in $P_6(p,q)$, then u precedes v in $P_6(p,q)$. We say that u is the *source* of the edge and that v is the *target* of the edge.

Let cde be a subpath of $P_t(p,q)$ such that $c,d \in B_{pq}(p)$ and the edge $[c,d]$ is at the i-th level of recursion of the call to FINDPATHREC(p, a, A) in FINDPATH(p, q), i.e. $|cd| \leq (2/s)^i|pq|$. Consider the set T_i of edges $[v,w]$ such that $[v,w]$ is an edge of $P_6(p,q)$ and the target w is in $B_{de}(d)$ but not in $B_{cd}(c)$. We claim that there can be at most 2 such edges and the sum of the lengths of the edges in T_i is at most $2(2/s)^i|pq|$, i.e.

$$\sum_{[v,w]\in T_i} |vw| \leq 2\left(\frac{2}{s}\right)^i |pq|.$$

If we sum up the lengths of all edges $[v,w]$ from level 1 to a maximum recursion depth m, we get that the length of the path from p to the representative a of $B_{pq}(p)$ is at most

$$\sum_{i=1}^{m} \sum_{[v,w]\in T_i} |vw| \leq \sum_{i=1}^{m} 2\left(\frac{2}{s}\right)^i |pq| \leq \sum_{i=1}^{\infty} 2\left(\frac{2}{s}\right)^i |pq| = \frac{4}{s-2}|pq|.$$

We now prove our claims. If T_i is empty, then the sum is zero. Otherwise, let an edge $[w_{j-1}, w_j]$ of $P_6(p,q)$ in T_i. From Lemma 7, we get $|w_{j-1}w_j| \leq (2/s)^i|pq|$ since the edge $[w_{j-1}, w_j]$ is in $B_{de}(d)$. We consider two cases: either (1) w_j is the representative of $B_{de}(d)$ or (2) it is not.

1. Suppose that w_j is the representative of $B_{de}(d)$, i.e. $w_j = d$.

   Consider the edge $[w_{j-2}, w_{j-1}]$ which precedes $[w_{j-1}, w_j]$ in $P_6(p,q)$. We consider two subcases: either (a) w_{j-1} is in $B_{cd}(c)$ or (b) it is not.

   (a) Suppose that w_{j-1} is in $B_{cd}(c)$.

   Therefore, only $[w_{j-1}, w_j]$ has its target in $B_{de}(d)$ and $|w_{j-1}w_j| \leq (2/s)^i|pq| \leq 2(2/s)^i|pq|$. Notice that, in this case, w_{j-1} is the representative of $B_{cd}(c)$ (thus $w_{j-1} = c$) because $w_{j-1} = c$ can only belong to one pair separating it from $w_j = d$.

   (b) Suppose that w_{j-1} is not in $B_{cd}(c)$.

   Since $w_j = d$, w_{j-1} must be strictly inside $B_{de}(d)$. The point w_{j-2} must be in $B_{cd}(c)$ since if it is outside $B_{cd}(c)$ but inside $B_{de}(d)$, then by Lemma 8, there is an edge from w_{j-2} to d which contradicts the existence of w_{j-1}. Furthermore, w_{j-2} is not the representative of $B_{cd}(c)$ since this would also contradict the existence of w_{j-1}. Therefore, the sum of the lengths of all edges having their target in $B_{de}(d)$ is $|w_{j-2}w_{j-1}| + |w_{j-1}w_j| \leq 2(2/s)^i|pq|$.

2. Suppose that w_j is not the representative of $B_{de}(d)$.

   From Lemma 8, we get that w_{j-1} must be in $B_{cd}(c)$ but not the representative of $B_{cd}(c)$. Otherwise, this would contradict the existence of w_j. Since w_{j-1} is in $B_{cd}(c)$, there is no other edge $[w_{k-1}, w_k]$, $k < j$, of $P_6(p,q)$ preceding $[w_{j-1}, w_j]$, where w_k is in $B_{de}(d)$ but not in $B_{cd}(c)$.

   Now, consider the edge $[w_j, w_{j+1}]$ which follows $[w_{j-1}, w_j]$ in $P_6(p,q)$. We consider two subcases: either (a) w_{j+1} is the representative of $B_{de}(d)$ or (b) it is not.

   (a) Suppose that w_{j+1} is the representative of $B_{de}(d)$.

   From Lemma 7, we get $|w_jw_{j+1}| \leq (2/s)^i|pq|$. Therefore, the sum of the lengths of all edges having their target in $B_{de}(d)$ is $|w_{j-1}w_j| + |w_jw_{j+1}| \leq 2(2/s)^i|pq|$.

   (b) Suppose that w_{j+1} is not the representative of $B_{de}(d)$.

From Lemma 8, we get that w_j has an edge to d. Because w_{j+1} is not the representative of $B_{de}(d)$, w_{j+1} must be outside of $B_{de}(d)$. Therefore, only $[w_{j-1}, w_j]$ has its target in $B_{de}(d)$ and not in $B_{cd}(c)$ and $|w_{j-1}w_j| \leq (2/s)^i|pq| \leq 2(2/s)^i|pq|$.

These cases cover all possibilities of edges in T_i. □

Theorem 5. *In an RSW-Spanner, the routing ratio of Algorithm 6 is at most* $\frac{4}{s} + \frac{6}{s-2} + 1$.

Proof. Let p and q be any two points. Let a be the representative of $B_{pq}(p)$, and let b be the representative of $B_{pq}(q)$. Let P_{pa} be the subpath from p to a of $P_6(p,q)$ and P_{bq} be the subpath from b to q of $P_6(p,q)$. From Lemma 13 and 14, we get:

$$|P_{pa}| + |ab| + |P_{bq}| \leq \frac{4}{s-2}|pq| + \left(1 + \frac{4}{s}\right)|pq| + \frac{2}{s-2}|pq| = \left(\frac{4}{s} + \frac{6}{s-2} + 1\right)|pq|.$$

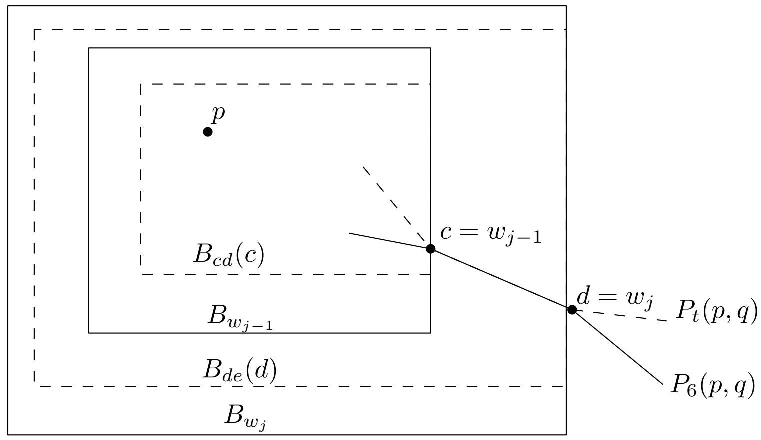
□

3.4 Improvement – 1-Local Routing Algorithm

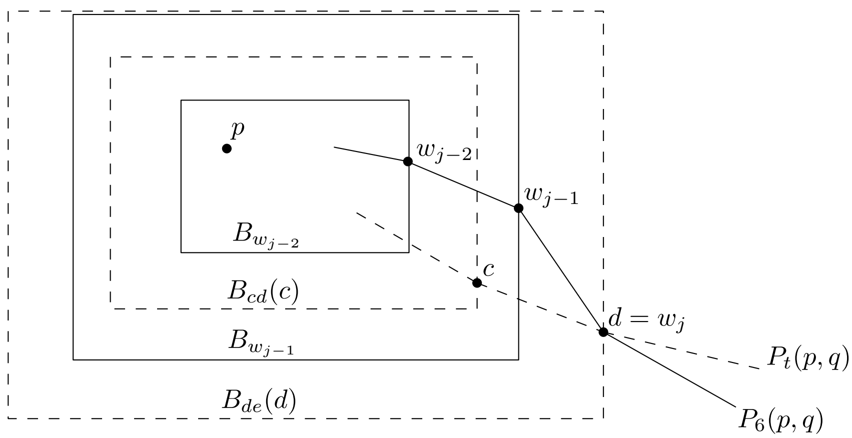
An important aspect of routing algorithms is how much information each point needs to store. In this section, we present an algorithm that is slightly different from Algorithm 6. The main difference is that it is 1-local instead of 2-local. Let v be the current point of the routing path. For all neighbors d of v, and for all neighbors e of d, in the previous section, we supposed that the following information was available at v:

- the edge $[v, d]$ together with $B_{vd}(v)$ and $B_{vd}(d)$;
- the edge $[d, e]$ together with $B_{de}(d)$ and $B_{de}(e)$.

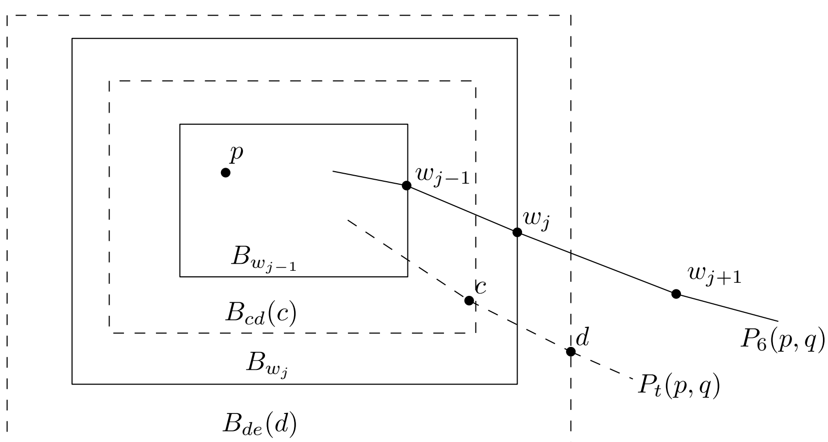
In this section, we explain how to design a routing algorithm that does not need to know the edge $[d, e]$ and the bounding boxes $B_{de}(d)$ and $B_{de}(e)$. However, it requires some other information about d. Recall that B_v is the largest bounding box that v is the representative of. Let $\mathcal{B}_d$ be the smallest bounding box that d is the representative



(a) An illustration of the case 1a



(b) An illustration of the case 1b (and 2a)



(c) An illustration of the case 2b)

Figure 11: Illustration of the cases of Lemma 14.

(a) The WSPD of the point set $\{v, d_1, d_2, q\}$.

(b) The bounding boxes of the split tree of the point set $\{v, d_1, d_2, q\}$ (except the one of the root of the split tree).

Figure 12: Illustration of a point set $\{v, d_1, d_2, q\}$ where v is the current point, $\mathcal{B}_{d_1}$ is not defined, and $\mathcal{B}_{d_2}$ is defined.

of and that contains B_v. Notice that d might not be the representative of a bounding box contains B_v. Thus, $\mathcal{B}_d$ might not be defined for some d. See Figure 12 for an illustration. Then, the following information is now available at v:

- the edge $[v, d]$ together with $B_{vd}(v)$ and $B_{vd}(d)$;
- the neighbor d of v together with the bounding box $\mathcal{B}_d$ if any.

As a result, this increases the upper bound on the routing ratio by $\frac{8}{s^2} + \frac{2}{s-2}$ (refer to Lemma 7). In our modified algorithm, only the Enlarging step differs from Algorithm 6 since the Reducing step in Algorithm 6 is already 1-local. This new algorithm does not necessarily find the representative a of $B_{pq}(p)$, but it finds a point a' that is the representative a bounding box $B_{a'q}(a')$. To find a path from p to some a', we use the following modified strategy. Let v be the current point on the path from p to a point a' that is the representative of a box $B_{a'q}(a')$ produced by our algorithm (at the beginning $v = p$). Our new algorithm selects the next edge in the following way. The distance between a circle C and a point c is defined by the smallest distance between the boundary of C and the point c, and is denoted by $|Cc|$. Let $\mathcal{V} = \{v' \in \mathcal{N}(v) \mid \mathcal{B}_{v'} \text{ is defined}, |C_{v'}q| \geq s\rho_{v'}\}$, where $\rho_{v'}$ is the radius of the enclosing circle $C_{v'}$ of $\mathcal{B}_{v'}$, and s is the separation ratio of the WSPD. Then, the next edge chosen by our algorithm is the edge $[v, w]$ such that the size of $\mathcal{B}_w$ is maximized among all $w \in \mathcal{V}$.

The strategy to find a path from b to q stays the same. Algorithm 7 below outlines the modified algorithm.

Algorithm 7 FINDPATHONELOCAL(v, p, q)

Input: the current point v,
the source p,
the destination q.

Output: The next point w on the path.

if there is an edge $[v, v']$ where v' is the representative of $B_{vq}(q)$ **then** // Reducing step
    $w \leftarrow v'$
else// Enlarging step
    $\forall v' \in \mathcal{N}(v)$, let $\mathcal{B}_{v'}$ be the smallest bounding box that v' is the representative of and that contains B_v, if any.
    Let $\mathcal{V} = \{v' \in \mathcal{N}(v) \mid \mathcal{B}_{v'} \text{ is defined}, |C_{v'}q| \geq s\rho_{v'}\}$
    $w \leftarrow \operatorname{argmax}_{v' \in \mathcal{V}} \operatorname{sizeof}(\mathcal{B}_{v'})$
end if
return w

Lemma 15. *In Algorithm 7, the amount of information stored at each point v_i is equal to $O(s^2 d(v_i)B)$, where B is the maximum number of bits to store a bounding box, and $d(v_i)$ is the number of neighbors of v_i. Moreover, $\sum_{i=1}^{n} O(s^2 d(v_i)B) = O(s^2 nB)$.*

Proof. Callahan and Kosaraju [14] proved that the number of pairs in a WSPD computed by the algorithm COMPUTEWSPD is $O(s^2 n)$. Each point in Algorithm 7 knows its neighbors and constant-size information about them. Since each set in a pair has only one representative, and the number of pairs is $O(s^2 n)$, the total size of the local information stored in all vertices is $O(s^2 nB)$ □

Notice that this new algorithm does not guarantee that the path stays inside $B_{pq}(p)$. However, as shown in the proof of Lemma 11, the endpoint of the first edge of the path from p to q that goes outside of $B_{pq}(p)$ has an edge to the representative of a bounding box containing q. Thus, Algorithm 7 is entering the Reducing step right after this edge is taken. We prove the correctness (refer to Theorem 6) and an upper bound on the routing ratio (refer to Theorem 7) of Algorithm 7.

Theorem 6. *Algorithm 7 finds a path in an RSW-Spanner from p to q. Moreover, let u be the current point the first time that the Reducing step is applied. The path taken from u to q is the same as the path found by the algorithm* FINDPATH*(u, q).*

Proof. We prove that of each edge $[v, w]$ taken in the Enlarging step of Algorithm 7 leads to the representative of a bounding box B_w that contains p and is larger than B_v. Furthermore, we prove that the source v of the edge $[v, w]$ is always inside $B_{pq}(p)$ in the Enlarging step. Then, we prove that Algorithm 7 enters the Reducing step which finds q. Thus, Algorithm 7 finds a path from p to q.

Recall that in Algorithm 7, we define $\mathcal{B}_{v'}$ as the smallest bounding box that v' is the representative of and that contains B_v, and

$$\mathcal{V} = \{v' \in \mathcal{N}(v) \mid \mathcal{B}_{v'} \text{ is defined}, |C_{v'}q| \geq s\rho_{v'}\}.$$

Suppose that the current point v is inside but not the representative of $B_{pq}(p)$. From Lemma 8, we get that v has an edge to the representative of a point of $P_t(p, q)$ that has a bounding box inside larger than B_v. This proves that there is always a choice of edges in the Enlarging step such that $\mathcal{B}_{v'}$ is defined and $|C_{v'}q| \geq s\rho_{v'}$. Therefore, when v is inside $B_{pq}(p)$, $\mathcal{V}$ is non-empty.

Let $P_7(p, q)$ be the path found by Algorithm 7 between two points p and q. Let $[r, u]$ be the first edge of the path $P_7(p, q)$ such that either:

- u is the representative of $B_{pq}(p)$; or
- r is inside but not the representative of $B_{pq}(p)$ and u is outside of $B_{pq}(p)$.

If u is the representative of $B_{pq}(p)$, then u has an edge to the representative of a bounding box containing q. Otherwise, since $B_{pq}(p) \subset \mathcal{B}_u$, by Lemma 6, u still has an edge to the representative of a bounding box containing q.

Let $\{U, Q\}$ be the pair separating u from q in the WSPD. Consider a call to FINDPATH(u, q). This call performs two call to FINDPATHREC: FINDPATHREC(u,

u, U) and FINDPATHREC(x, q, Q). The call to FINDPATHREC(u, u, U) terminates immediately. We show that the path taken from x to q in Algorithm 7 is the same as the path found by the call to FINDPATHREC(x, q, Q). By Lemma 4, since x is the representative of $B_{uq}(q)$, x is also the representative of $B_{xq}(x)$. Let y be the representative of $B_{xq}(q)$. Thus, the Reducing step takes the edge $[x, y]$. Furthermore, since x is the representative of $B_{xq}(x)$, the call to FINDPATHREC(x, q, Q) also takes the edge $[x, y]$. Then, both algorithms repeat this step until q is found. Thus, the path taken from x to q is the same as the path found by the call to FINDPATHREC(x, q, Q). □

Lemma 16. *Consider any RSW-Spanner. In Algorithm 7, the diameter of the last enclosing circle in the Enlarging step is at most* $(2/s)|pq|$.

Proof. Let C_w be the last enclosing circle in the Enlarging step, and ρ_w be the radius of C_w. Since p is in C_w, and $|C_w q| \geq s\rho_w$, we also know that $|pq| \geq s\rho_w$. Thus, we get $\rho_w \leq \frac{|pq|}{s}$, from which $2\rho_w \leq \frac{2}{s}|pq|$. □

Theorem 7. *In an RSW-Spanner, the routing ratio of Algorithm 7 is at most* $\frac{8}{s^2} + \frac{4}{s} + \frac{8}{s-2} + 1$.

Proof. Let $P_7(p, q)$ be the path found by Algorithm 7 between two points p and q. Let $[r, u]$ be the first edge of the path $P_7(p, q)$ such that either:

- u is the representative of $B_{pq}(p)$; or
- r is inside but not the representative of $B_{pq}(p)$ and u is outside of $B_{pq}(p)$.

If u is the representative of $B_{pq}(p)$, then u has an edge to the representative of a bounding box containing q. Otherwise, since $B_{pq}(p) \subset \mathcal{B}_u$, by Lemma 6, u still has an edge to the representative of a bounding box containing q. By Lemma 16, since $[r, u]$ is in $\mathcal{B}_u$, we get that $|ru| \leq (2/s)|pq|$.

Let cde be a subpath of $P_t(p, q)$ such that $c, d \in B_{pq}(p)$ and the edge $[c, d]$ is at the i-th level of recursion of the call to FINDPATHREC(p, a, A) in FINDPATH(p, q). Let T_i be the set of edges $[v, w]$ such that $[v, w]$ is an edge of $P_7(p, q)$ and the target w is in $B_{de}(d)$ but not in $B_{cd}(c)$. We prove that, for $i \geq 1$, T_i contains at most 2 edges and the sum of the lengths of the edges in T_i is at most $2(2/s)^i|pq|$, i.e.

$$\sum_{[v,w]\in T_i} |vw| \leq 2\left(\frac{2}{s}\right)^i |pq|.$$

If T_i is empty, then the sum is zero. Otherwise, let an edge $[w_{j-1}, w_j]$ of $P_7(p, q)$ be in T_i. From Lemma 7, we get $|w_{j-1}w_j| \leq (2/s)^i|pq|$ since the edge $[w_{j-1}, w_j]$ is in $B_{de}(d)$. We consider two cases: either (1) w_j is the representative of $B_{de}(d)$ or (2) it is not.

(1) Suppose that w_j is the representative of $B_{de}(d)$, i.e. $w_j = d$.

Consider the edge $[w_{j-2}, w_{j-1}]$ which precedes $[w_{j-1}, w_j]$ in $P_7(p, q)$. We consider two subcases: either (a) w_{j-1} is in $B_{cd}(c)$ or (b) it is not.

(a) Suppose that w_{j-1} is in $B_{cd}(c)$.

Therefore, only $[w_{j-1}, w_j]$ has its target in $B_{de}(d)$ and $|w_{j-1}w_j| \leq (2/s)^i|pq| \leq 2(2/s)^i|pq|$. Notice that, in this case, w_{j-1} is the representative of $B_{cd}(c)$ (thus $w_{j-1} = c$) because $w_{j-1} = c$ can only belong to one pair separating it from $w_j = d$.

(b) Suppose that w_{j-1} is not in $B_{cd}(c)$.

Since $w_j = d$, w_{j-1} must be strictly inside $B_{de}(d)$. The point w_{j-2} must be in $B_{cd}(c)$ since if it is outside $B_{cd}(c)$ but inside $B_{de}(d)$, then by Lemma 8, there is an edge from w_{j-2} to d which contradicts the existence of w_{j-1}. Furthermore, w_{j-2} is not the representative of $B_{cd}(c)$ since this would also contradict the existence of w_{j-1}. Therefore, the sum of the lengths of all edges having their target in $B_{de}(d)$ is $|w_{j-2}w_{j-1}| + |w_{j-1}w_j| \leq 2(2/s)^i|pq|$.

(2) Suppose that w_j is not the representative of $B_{de}(d)$.

From Lemma 8, we get that w_{j-1} must be in $B_{cd}(c)$ but not the representative of $B_{cd}(c)$. Otherwise, this would contradict the existence of w_j. Since w_{j-1} is in $B_{cd}(c)$, there is no other edge $[w_{k-1}, w_k]$, $k < j$, of $P_6(p, q)$ preceding $[w_{j-1}, w_j]$, where w_k is in $B_{de}(d)$ but not in $B_{cd}(c)$.

Now, consider the edge $[w_j, w_{j+1}]$ which follows $[w_{j-1}, w_j]$ in $P_7(p,q)$. We consider two subcases: either (a) w_{j+1} is the representative of $B_{de}(d)$ or (b) it is not.

(a) Suppose that w_{j+1} is the representative of $B_{de}(d)$.

From Lemma 7, we get $|w_j w_{j+1}| \leq (2/s)^i |pq|$. Therefore, the sum of the lengths of all edges having their target in $B_{de}(d)$ is $|w_{j-1}w_j| + |w_j w_{j+1}| \leq 2(2/s)^i|pq|$.

(b) Suppose that w_{j+1} is not the representative of $B_{de}(d)$.

From Lemma 8, we get that w_j has an edge to d. Because w_{j+1} is not the representative of $B_{de}(d)$, w_{j+1} must be outside of $B_{de}(d)$. Therefore, only $[w_{j-1}, w_j]$ has its target in $B_{de}(d)$ and not in $B_{cd}(c)$ and $|w_{j-1}w_j| \leq (2/s)^i|pq| \leq 2(2/s)^i|pq|$.

These cases cover all possibilities of edges in T_i.

Consider the set T_1. Notice that T_1 is the set of edges $[v,w]$ such that $[v,w]$ is an edge of $P_7(p,q)$ and the target w is in $B_{pq}(p)$ but not in $B_{pa}(p)$, where a is the representative of $B_{pq}(p)$. Let $\mathcal{T} = \{[r,u]\} \cup T_1$. We prove that $\mathcal{T}$ contains at most 2 edges and the sum of the lengths of the edges in $\mathcal{T}$ is at most $2(2/s)|pq|$. Recall that $[r,u]$ is the first edge of the path $P_7(p,q)$ such that u is the representative of $B_{pq}(p)$, or r is in $B_{pq}(p)$ but not the representative of $B_{pq}(p)$ and u is outside of $B_{pq}(p)$. We have two cases: (1) u is the representative of $B_{pq}(p)$; (2) r is in $B_{pq}(p)$ but not the representative of $B_{pq}(p)$ and u is outside of $B_{pq}(p)$.

(1) Suppose that u is the representative of $B_{pq}(p)$.

Then, the edge $[r,u]$ is in T_1, and $\mathcal{T} = T_1$. Thus,

$$\sum_{[v,w]\in\mathcal{T}} |vw| = \sum_{[v,w]\in T_1} |vw| \leq 2(2/s)|pq|.$$

(2) Suppose that r is in $B_{pq}(p)$ but not the representative of $B_{pq}(p)$ and u is outside of $B_{pq}(p)$.

If r is in $B_{pa}(p)$, then T_1 is empty and $\mathcal{T}$ only contains the edge $[r,u]$ of length at most $(2/s)|pq|$. Otherwise, consider the point r' preceding r in $P_7(p,q)$. By

Lemma 8, if r' was in $B_{pq}(p)$ but not in $B_{pa}(p)$, then r' would have an edge to a which would contradict the existence of r. Thus, r' must be in $B_{pa}(p)$. Therefore, the edge $[r', r]$ is the only edge in T_1, and the length of $[r', r]$ is at most $(2/s)|pq|$ by Lemma 1 since $[r', r]$ is in $B_{pq}(p)$. Then, $\mathcal{T} = \{[r', r], [r, u]\}$ and $|r'r| + |ru| \leq 2(2/s)|pq|$.

If we sum up the lengths of all edges $[v, w]$ from level 2 to a maximum depth m, and the lengths of the edges in $\mathcal{T}$, we get that the length of the path found in the Enlarging step is at most

$$\begin{aligned}
&\sum_{[v,w]\in\mathcal{T}} |vw| + \sum_{i=2}^{m} \sum_{[v,w]\in T_i} |vw| \\
\leq\ & 2\left(\frac{2}{s}\right)|pq| + \sum_{i=2}^{m} \sum_{[v,w]\in T_i} |vw| \\
\leq\ & 2\left(\frac{2}{s}\right)|pq| + \sum_{i=2}^{m} 2\left(\frac{2}{s}\right)^i |pq| \\
=\ & \sum_{i=1}^{m} 2\left(\frac{2}{s}\right)^i |pq| \\
\leq\ & \sum_{i=1}^{\infty} 2\left(\frac{2}{s}\right)^i |pq| \\
=\ & \frac{4}{s-2}|pq|
\end{aligned}$$

Since u has an edge to a bounding box containing q, Algorithm 7 enters the Reducing step. We bound the path found by Algorithm 7 by comparing its length to the distance $|uq|$. By Theorem 6, we know that the path taken from u to q is the same as the path found by the algorithm FindPath(u, q). Let $\{U, Q\}$ be the pair separating u from q in the WSPD. Consider the call to FindPath(u, q). This call performs two call to FindPathRec: FindPathRec(u, u, U) and FindPathRec(x, q, Q). The call to FindPathRec(u, u, U) terminates immediately. By Lemma 1, the length of the edge $[u, x]$ is at most $(1 + 4/s)|uq|$. In the proof of Theorem 2, we show that the length of the path found by FindPathRec(x, q, Q) is at most $(2/(s-2))|uq|$. Thus, the length of the path found by Algorithm 7 from u to q is at most $|ux| + (2/(s-2))|uq| \leq (1 + 4/s + 2/(s-2))|uq|$.

Since the diameter of the enclosing circle of $\mathcal{B}_u$ is at most $(2/s)|pq|$ from Lemma 16, and since p is in $\mathcal{B}_u$, we have $|up| \leq (2/s)|pq|$. By the triangle inequality, we get that $|uq| \leq |up| + |pq| \leq (2/s)|pq| + |pq| = (1 + 2/s)|pq|$. Let P_{pu} be the subpath from p to u of $P_7(p, q)$ and P_{uq} be the subpath from u to q of $P_7(p, q)$. We then get that the length of the path is at most

$$
\begin{aligned}
& |P_{pu}| + |P_{uq}| \\
\leq \\
& \frac{4}{s-2}|pq| + \left(1 + \frac{4}{s} + \frac{2}{s-2}\right)|uq| \\
\leq \\
& \frac{4}{s-2}|pq| + \left(1 + \frac{4}{s} + \frac{2}{s-2}\right)\left(1 + \frac{2}{s}\right)|pq| \\
= \\
& \left(\frac{8}{s^2} + \frac{4}{s} + \frac{8}{s-2} + 1\right)|pq|.
\end{aligned}
$$

$\square$

Chapter 4

Conclusion

In this chapter, we summarize the results of this thesis and explore directions for future research.

4.1 Summary of Results

Our main contribution is a competitive local-routing algorithm on a WSPD-spanner that we called RSW-Spanner, with a near-optimal routing ratio, $1 + O(1/s)$. Given a pointset and a separation ratio s, a WSPD with separation ratio s is (typically) not unique. We based the construction of the WSPD on the split tree data structure. From this WSPD, we showed how to construct the RSW-Spanner that facilitates local routing by selecting a well-chosen edge from each partition rather than picking an arbitrary edge in general WSPD-spanners. As a side benefit, the RSW-Spanner has a slightly improved spanning ratio, $1+4/s+4/(s-2)$, over the original one, $1+8/(s-4)$. This improvement stems from the additional properties our well-chosen edges have. On the RSW-Spanner, we presented a 2-local and a 1-local routing algorithm with competitive routing ratios of $1 + 4/s + 6/(s - 2)$ and $1 + 4/s + 8/(s - 2) + 8/s^2$, respectively. Ideally, one would like the routing ratio to be identical to the spanning ratio, however, this is rarely the case when routing locally since an adversary can often force an algorithm to stray from the actual shortest path. We proved a lower bound of $1 + 8/s$ on the spanning ratio of the RSW-Spanner, thereby proving the near-optimality of the spanning ratio of the RSW-Spanner and the near-optimality of the routing ratio of both our routing algorithms.

4.2 Future Work

This thesis mostly addressed the problem of local routing in 2-dimensional spanners. WSPDs can be defined in $\mathbb{R}^d$ in general. We think that the results of this thesis extend to higher dimensions. Also, as said in Section 2.3, there exist WSPD-based spanners with bounded degree and bounded diameter. Doing local routing in spanners with bounded degree would be interesting in that it would decrease the size of the local information at each point. As for spanners with bounded diameter, it would be interesting to do local routing in those such that the number of edges in the path be comparable to the diameter.

WSPDs have been used before as an aid to routing in unit-disk graphs by Kaplan et al. [19]. They showed a local routing algorithm on the unit-disk graph that uses a WSPD constructed with the unit-disk graph as a metric. Moreover, there are several versions of WSPD based on the data structure that is used in their construction. For example, the WSPD we used in this thesis was constructed using split trees. Sariel Har-Peled [18] showed how to construct WSPDs with quadtrees. We think that the results of this thesis extend to quadtrees and other similar types of trees. Thus, by exploring the use of other data structures and other metrics, it would be interesting to search for spanners in high-dimensions where it is possible to do local routing with a better routing ratio.

More generally, another well-known family of spanners possible in high-dimensions are greedy spanners [2, 3] derived from Kruskal's algorithm. They are constructed by considering each edge in non-decreasing order, checking if there is a path satisfying the requirements of the t-spanner, and if not, adding the edge to the graph. No local routing algorithm is known for this type of graph.

List of References

[1] P. K. Agarwal, H. Edelsbrunner, and O. Schwarzkopf. Euclidean minimum spanning trees and bichromatic closest pairs. *Discrete Comput. Geom.*, 6:407–422, 1991.

[2] S. P. A. Alewijnse, Q. W. Bouts, A. P. ten Brink, and K. Buchin. Computing the greedy spanner in linear space. *Algorithmica*, 73(3):589–606, 2015.

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

[4] S. Arya, G. Das, D. M. Mount, J. S. Salowe, and M. H. M. Smid. Euclidean spanners: short, thin, and lanky. In *Symp. Theory Comput. (STOC)*, pages 489–498, 1995.

[5] S. Arya, D. M. Mount, and M. H. M. Smid. Randomized and deterministic algorithms for geometric spanners of small diameter. In *Symp. Found. Comput. Sci. (FOCS)*, pages 703–712, 1994.

[6] N. Bonichon, P. Bose, J.-L. De Carufel, L. Perkovic, and A. van Renssen. Upper and lower bounds for online routing on Delaunay triangulations. In *European Symp. Algorithms (ESA)*, pages 203–214, 2015.

[7] P. Bose, P. Carmi, S. Collette, and M. H. M. Smid. On the stretch factor of convex delaunay graphs. *JoCG*, 1(1):41–56, 2010.

[8] P. Bose, P. Carmi, and S. Durocher. Bounding the locality of distributed routing algorithms. *Distrib. Comput.*, 26(1):39–58, 2013.

[9] P. Bose, L. Devroye, M. Löffler, J. Snoeyink, and V. Verma. Almost all Delaunay triangulations have stretch factor greater than pi/2. *Comput. Geom.*, 44(2):121–127, 2011.

[10] P. Bose, R. Fagerberg, A. van Renssen, and S. Verdonschot. Optimal local routing on Delaunay triangulations defined by empty equilateral triangles. *SIAM J. Comput.*, 44(6):1626–1649, 2015.

[11] P. Bose and P. Morin. Online routing in triangulations. *SIAM J. Comput.*, 33(4):937–951, 2004.

[12] P. B. Callahan and S. R. Kosaraju. Faster algorithms for some geometric graph problems in higher dimensions. In *Symp. Discrete Algorithms (SODA)*, pages 291–300, 1993.

[13] P. B. Callahan and S. R. Kosaraju. Algorithms for dynamic closest pair and n-body potential fields. In *Symp. Discrete Algorithms (SODA)*, pages 263–272, 1995.

[14] P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. ACM*, 42(1):67–90, 1995.

[15] P. Chew. There is a planar graph almost as good as the complete graph. In *Symp. Comput. Geom. (SOCG)*, pages 169–177, 1986.

[16] P. Chew. There are planar graphs almost as good as the complete graph. *J. Comput. Syst. Sci.*, 39(2):205–219, 1989.

[17] D. P. Dobkin, S. J. Friedman, and K. J. Supowit. Delaunay graphs are almost as good as complete graphs. *Discrete Comput. Geom.*, 5:399–407, 1990.

[18] S. Har-Peled. *Geometric Approximation Algorithms.* American Mathematical Society, Boston, MA, USA, 2011.

[19] H. Kaplan, W. Mulzer, L. Roditty, and P. Seiferth. Routing in unit disk graphs. In *Lat. Am. Symp. Theor. Inform. (LATIN)*, pages 536–548, 2016.

[20] J. M. Keil and C. A. Gutwin. Classes of graphs which approximate the complete euclidean graph. *Discrete Comput. Geom.*, 7:13–28, 1992.

[21] S. Misra, I. Woungang, and S. C. Misra. *Guide to Wireless Sensor Networks.* Springer Publishing Company, Incorporated, 1st edition, 2009.

[22] G. Narasimhan and M. H. M. Smid. *Geometric spanner networks.* Cambridge University Press, 2007.

[23] G. Xia. The stretch factor of the Delaunay triangulation is less than 1.998. *SIAM J. Comput.*, 42(4):1620–1659, 2013.