RC24988 (W1005-018) May 3, 2010
Computer Science

IBM Research Report

Connectivity Graphs of Uncertain Regions

Erin Chambers[1], Alejandro Erickson[2], Sándor Fekete[3], Jon Lenchner[4], Jeff Sember[5], Venkatesh Srinivasan[6] Ulrike Stege[6], Svetlana Stolpner[5], Christophe Weibel[5], Sue Whitesides[6]

[1]Saint Louis University

[2]Simon Fraser University

[3]Technische Universität, Braunschweig

[4]IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

[5]University of British Columbia

[6]University of Victoria

[7]McGill University

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

Connectivity graphs of uncertain regions*

Erin Chambers, Alejandro Erickson, Sándor Fekete, Jon Lenchner,
Jeff Sember, Venkatesh Srinivasan, Ulrike Stege, Svetlana Stolpner,
Christophe Weibel, and Sue Whitesides

Abstract. In this paper we provide an algorithmic study of connectivity problems with neighborhoods: Given a family of geometric sets, choose one point per set, such that the length of the longest edge in a connecting structure is minimized. We show that even simple geometric scenarios give rise to problems that are NP-hard to approximate; on the positive side, we give approximation and exact algorithms.

1 Introduction

Finding an optimal connected structure is one of the fundamental optimization problems in network design. The standard problem of minimizing the total edge cost in the network amounts to a minimum spanning tree, which can be computed by straightforward greedy methods. A closely related problem that has gained in importance in the context of wireless networking is to consider the "bottleneck" problem of minimizing the length of the longest edge, which corresponds to choosing the necessary power and thus range for the routers to be placed at nodes. Quite conveniently, the same greedy methods still yield optimal solutions.

The situation changes when the location of routers becomes part of the problem: How should each location be chosen from a given neighborhood, such that the solution of the resulting connectivity problem is as good as possible? The neighborhoods can be the result of imprecise input data, or simply arise from a geometric range of possible locations; depending on the scenario, the choice of locations can be optimistic (i.e., best case) or adversarial (i.e., worst case).

Formally, our problem can be defined as follows. Let U denote a family of uncertainty regions, e.g., a family of disks, squares, line segments or pairs of points. For each uncertainty region $u_i \in U$, one point p_i is to be chosen inside this region. Let P be the set of points chosen. For some value $\alpha \in \mathbb{R}$, we define the *connectivity graph* $G_\alpha = (V, E)$ of P with respect to α as follows: $V = P$ and $E = \{(p_i \in P, p_j \in P), \|p_i - p_j\|_2 \leq 2\alpha\}$. In other words, the graph connects a pair of points with an edge whenever closed disks of radius α centered at these points intersect. Motivated by the two scenarios described above, we define the following two problems.

1. **Problem** $B\alpha$ (Best Case α) Find the minimum value α such that there exists a set of points P such that the connectivity graph G_α of P is connected.
2. **Problem** $W\alpha$ (Worst Case α) Find the minimum value α such that for any choice of points P the connectivity graph G_α of P is connected.

We study problems $B\alpha$ and $W\alpha$ for different kinds of uncertainty regions.

* The authors are grateful for two Bellairs workshops supporting this research: the 8th and 9th McGill—INRIA Workshop on Computational Geometry in 2009 and 2010.

Related Work. If the n uncertainty regions are points (in other words there is no uncertainty), an efficient algorithm is known for finding the minimum α for which the connectivity graph is connected. The algorithm, due to Delfinado and Edelsbrunner [5], has a running time of $O(n\alpha(n))$, where $\alpha(n)$ denotes the inverse of the Ackermann function.

The well-studied family of range assignment problems is closely related. In these problems the disks centered at each point can be of different radii, and the goal is to minimize the total power consumption under the constraint that the network satisfies certain structural properties like connectivity, strong connectivity, or a particular broadcast property. Most of the work on these problems has considered point sets rather than uncertainty regions (see [3, 12, 13, 1, 9]).

The minimum spanning tree problem has been studied in the setting of uncertainty regions. Yang *et al.* [18] showed that the problem of computing a spanning tree that minimizes the total edge length is NP-hard if the uncertainty regions are non-overlapping unit disks or rectangles. They also give a polynomial-time approximation scheme (PTAS) for the case where the uncertainty regions are unit disks; this is notably different from our problem, which does not allow a PTAS, unless P=NP. Other optimization problems with neighborhoods that have received quite some attention include the Traveling Salesman Problem; e.g. see [2, 11, 8, 4, 15]. The bottleneck version of TSP is known to be NP-hard [10, p. 212]. A 2-approximation has been known since 1984 [16] which, interestingly, only requires TSP with the triangle inequality.

Our Main Results. After sketching that many variants of the $B\alpha$ problem are NP-hard (some even to approximate), we give positive algorithmic results for for the $B\alpha$ problem for several types of uncertainty regions As the underlying problem is inherently geometric in nature, we focus our attention on Euclidean scenarios. Our main results are as follows:

1. We show that the $B\alpha$ problem is NP-hard when the uncertainty regions are all squares, vertical line segments, and pairs of points; we show that it is NP-hard to approximate $B\alpha$ within a factor less than $\sqrt{5}/2$ when the uncertainty regions are pairs of points. See Section 2.
2. For uncertainty regions which are all unit disks, we show constant additive approximation algorithms for $W\alpha$ and $B\alpha$. The algorithm for $W\alpha$ is shown to be a constant (multiplicative) factor approximation while the $B\alpha$ algorithm is not. However, a slight modification of the $B\alpha$ algorithm gives a constant multiplicative approximation in case the disks are *non-overlapping*. See Section 3.
3. We present an exact algorithm for $B\alpha$ when the instance consists of $n - k$ fixed points and k line segments. The algorithm is polynomial in n for constant k. See Section 4.

2 Hardness Results

In this paper, we prove two hardness results for variants of the Bα problem. Our first theorem shows NP-hardness when the uncertainty regions are non-overlapping point pairs. This result also implies a hardness of approximation result for the case of point pairs and a NP-hardness result for the case of line segments (see Section 6.2 and 6.3 in the appendix). Our second theorem proves NP-hardness for the case of non-overlapping uncertain square regions. In this section, we sketch the main ideas behind the first result. The full proofs of both the results can be found in the appendix.

NP-hardness of $B\alpha$ for line segments and point pairs. We consider the $B\alpha$ problem for the simple case of non-overlapping uncertainty regions of vertically aligned pairs of points, unit distance apart with integer coordinates. We study the decision version of $B\alpha$ problem for $\alpha = 1$, *i.e.*, we want to decide if G_1 is connected

for some choice of points, one for each uncertainty pair. By using a reduction from Planar 3-SAT, we will show that this problem in NP-hard.

Overview of the Reduction. In our proof, we show a reduction from Planar 3-SAT, 3-SAT with the added condition that the input formula can be represented as a planar graph. Formally, Planar 3-SAT is defined as follows. Let $\Phi = (X, C)$ be an instance of 3-SAT, with variable set $X = \{x_1, \ldots, x_n\}$ and clauses $C = \{c_1, \ldots, c_m\}$. Each clause consists of exactly three literals, each a variable or its negation. For such an instance, we define a formula graph $H(\Phi)$ as follows: $H(\Phi) = (V, E)$ with vertex set $V = X \cup C$ and edge set $E = E_1 \cup E_2$, such that $E_1 = \{(x_i, x_{i+1}) | i < n\}$, and $E_2 = \{(x_i, c_j) | c_j \text{ contains } x_i \text{ or } \overline{x_i}\}$. A planar 3-SAT instance is one whose corresponding formula graph $H(\Phi)$ is planar. The Planar 3-SAT problem is to determine whether a planar 3-SAT instance Φ is satisfiable. Planar 3-SAT problem is known to be NP-complete [14].

We make use of the fact that, given a planar 3-SAT instance Φ with formula graph $H(\Phi)$, this graph has a planar layout on a $O(n) \times O(n)$ grid [7, 17]. Further, in this layout, the vertices (variables and clauses) can be drawn as horizontal line segments and edges as vertical line segments.

To reduce from Planar 3-SAT to an instance of $B\alpha$, where the uncertainty regions are pairs of points, we make use of a number of gadgets. Specifically, given a layout of a Planar 3-SAT instance using line segments as described above, we replace each horizontal line segment corresponding to a variable by a variable gadget, each horizontal line segment corresponding to a clause by a clause gadget, and each edge by an appropriate vertical sequence of uncertainty pairs. We will argue that there exists a choice of points in each of these uncertainty pairs such that the connectivity graph for $\alpha = 1$, G_1, is connected if and only if the corresponding Planar 3-SAT instance is satisfiable. We next describe the different gadgets in detail and how they may be connected.

Overview of the Gadgets. We give the main ideas behind the three types of gadgets we use in our reduction: clause gadget, variable gadget and connectors linking the gadgets.

A clause gadget is designed so that it contains three "gates", one for each of the literals in the clause. The gate for each literal will be either on the top or the bottom of the clause gadget depending on whether the literal appears below or above the clause in the planar layout. For the connectivity subgraph corresponding to the clause to be connected to the rest of the graph in G_1, at least one of these three gates should be open. This will correspond to setting the literal to "true". This, in turn, ensures that the clause is satisfied. See Figure 5 in the Appendix.

The role of a variable gadget is to choose and propagate a truth value for the variable to all the clauses containing it in a consistent manner. The variable gadget contains three types of constructs. Type I and type II constructs will help link the variable to all the clauses that contain it and are either above or below it. We have one such type I-type II pair for every occurence of the variable in a clause. A construct of type III is used to ensure that the truth assignment to the variable in all the copies of type I-type II pair are the same. If not, the graph G_1 corresponding to this variable gadget is not connected. See Figure 7 in the Appendix.

The variable and clause gadgets are linked to each other using two types of connectors. The connector linking a clause to a variable ensures a consistent assignment of truth value to a variable and a clause that contains this variable or its negation. Inconsistent assignments will result in G_1 being disconnected. The connector linking a variable to another variable can be more flexible. It should help connect one variable gadget to another variable gadget in G_1 irrespective of the choice of truth values for each of them. See Figure 8 in the Appendix.

3 Constant Factor and Additive Approximations

Our approximations are based on computing minimum bottleneck spanning trees, MBSTs, which are spanning trees that minimize the maximum length edge in the tree. These can be computed in $O(n\alpha(n))$ time the methods of [5].

Lemma 1. *Given a set of uncertainty regions which are unit disks $D_1, \ldots, D_n$ with centers $p_1, \ldots, p_n$, let L be the largest edge of a bottleneck shortest path tree on $\{p_i\}$. Then choosing broadcasting locations $\ell_i = p_i$ and $\alpha = L/2$ is at worst an OPT+1 approximation to the $B\alpha$ Problem. In other words, if OPT denotes the smallest radius α for any choice of $\ell_i \in D_i$, then $L/2 \leq$ OPT+1. We can compute this approximation in time $O(n\alpha(n))$.*

Proof. Let L be the maximum length edge of a MBST on $\{p_i\}$. Consider the best choice of the $\ell_i \in D_i$ and an associated MBST on these ℓ_i. The edges of this MBST are each at most 2 shorter than the corresponding edges of a spanning tree, S, on the corresponding p_i. Thus the maximum length of any edge in S is at most 2 greater than the maximum length edge in the MBST on the ℓ_i, and so too the maximum length, L, of any edge of a MBST on the $\{p_i\}$ must also be at most 2 greater than the maximum length edge in the MBST on the ℓ_i. The result follows.

Lemma 2. *Given a set of uncertainty regions which are unit disks $D_1, \ldots, D_n$ with centers $p_1, \ldots, p_n$, let L be the largest edge of a MBST on $\{p_i\}$. Then choosing $r = L/2 + 1$ always results in connectivity graph which is connected and is at worst an OPT+1 approximation to the $W\alpha$ Problem - Problem 2.*

Proof. First note that the connectivity graph given by any selection of $\ell_i \in D(p_i; 1)$ and $\{D(\ell_i; L/2 + 1)\}$ is connected since if (p_i, p_j) is an edge of a MBST on $\{p_k\}$ then $D(\ell_i; L/2 + 1) \cap D(\ell_j; L/2 + 1) \neq \emptyset$ for any choice of $\ell_i \in D(p_i; L/2 + 1), \ell_j \in D(p_j; L/2 + 1)$. We are thus left to show that choosing $r = L/2 + 1$ is at worst an OPT+1 approximation. But clearly we can choose $\ell_i = p_i \ \forall i$ whence the minimum α is $L/2$. Hence OPT $\geq L/2$ and the lemma is established.

Lemma 3. *Given a set of uncertainty regions which are unit disks $D_1, \ldots, D_n$ with centers $p_1, \ldots, p_n$, let L be the largest edge of a MBST on $\{p_i\}$. Then choosing $\alpha = L/2 + 1$ is at worst a factor 2 approximation to OPT for the $W\alpha$ Problem.*

Proof. Note that OPT+1 $\leq$ 2OPT as long as OPT ≥ 1 so, by Lemma 2, it suffices to show that OPT ≥ 1. By assumption there are more than one disk centers $\{p_i\}$. Let p_j be a leftmost point amongst the $\{p_i\}$ and choose ℓ_j to be the leftmost point in $D(p_j; 1)$ and for all other $D(p_k; 1)_{k \neq j}$ choose ℓ_k to be the rightmost point in $D(p_k; 1)$. Then ℓ_j is at least distance 2 from each of the other ℓ_k and so we must choose $\alpha \geq 1$ to keep the alpha complex connected. It follows that OPT ≥ 1 and the lemma is established.

Unfortunately our approximation for the $B\alpha$ Problem is not a constant factor approximation, since if one takes n unit disks with non-empty intersection, then the ℓ_i can all be taken to equal one of the intersection points so that OPT$= 0$ while $L/2$ can be non-zero (and as big as 1).

In the case of the $W\alpha$ Problem, Lemma 3 shows that the simple broadcast-from-center heuristic can be no worse than a 2-approximation. However, we have found examples showing only that the approximation can be (asymptotically) as bad as a $\sqrt{2}$-approximation, as the next Lemma asserts. The proof, which amounts to a detailed construction, is given in the Appendix.

Lemma 4. *Given any $L > 2$, there is an instance of the $W\alpha$ Problem with uncertainty regions that are unit disks $\{D_i = D_i(p_i; 1)\}_{i=1}^n$ such that the longest edge of a MBST on the $\{p_i\}$ is L and OPT is as small as $\frac{\sqrt{L^2+4}}{2}$, and therefore the algorithm of Lemma 2 can be as bad as a factor $\sqrt{2} - \epsilon$ approximation for arbitrarily small ϵ.*

While the broadcast-from-center heuristic is not a constant factor approximation to the $B\alpha$ Problem for highly overlapping unit disks, it is easy to make a small modification to the heuristic so that it *is* a constant factor approximation to $B\alpha$ for *non-overlapping* unit disks. A problem for the heuristic, as it stands, in the case

of non-overlapping disks occurs if we have just two disks and these two disks are within ϵ of being tangent to one another. As $\epsilon \to 0$ one can choose broadcast locations ℓ_i increasingly close together so broadcast-from-center becomes arbitrarily bad. However, we can either deal with two disks as a special case, or take the following more principled approach: begin as in broadcast-from-center by picking the centers of all uncertainty disks, and then finding a MBST on these centers, but at the end, "cinch-up" any leaf nodes by bringing the broadcast locations for these disks as close as possible to their parent nodes. In the case that the MBST is actually a simple path, be sure to cinch-up twice, first at one end, then at the other. This process ensures that we always obtain OPT for two disks. For three disks, we are not guaranteed to have OPT, but since points on three unit disks cannot come arbitrarily close to one another, the modified broadcast-from-center heuristic is a constant factor approximation. See Figure 1.

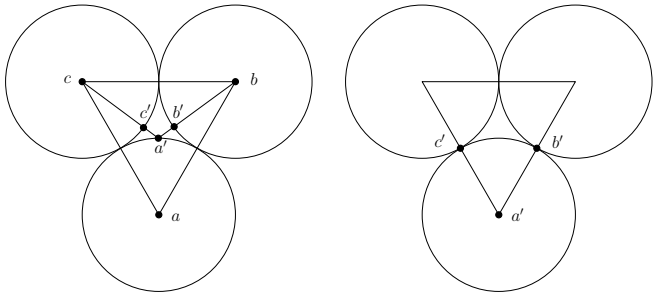

Fig. 1. The $B\alpha$ Problem for three (almost) tangent unit disks. OPT, as shown on the left is given by the choice of broadcasting locations (a', b', c'), not quite on an equilateral triangle, with MBST cost $(\sqrt{1+(\sqrt{3}-1)^2}-1)/2 \approx .12$, while the "cinch-up" heuristic, on right, chooses broadcasting locations (a', b', c') with cost .5.

4 An exact algorithm solving $B\alpha$ for n fixed points and k segments

In this section, we present an exact algorithm that solves variant k-$B^L\alpha$ of $B\alpha$. Here, given are a set of n fixed points and as uncertainty regions k line segments of any length and orientation. While it is not hard to see that $B\alpha$ is fixed-parameter tractable[1] for parameter k when the k uncertainty regions are pairs of points,[2] pains were taken to obtain an exact algorithm for k-$B^L\alpha$ that is polynomial in n for any fixed k. Our algorithm determines a set of point positions on the segments, such that there is a spanning tree connecting all the points on the segments as well as all fixed points, with no tree edge longer than L, where $L = 2\alpha$ and α is minimized. In other words, we seek to find a spanning tree connecting exactly one point of each segment and all fixed points, with its longest edges being of minimum length. We first show some properties w.r.t. the positions of points on segments in an optimal solution. Then we present our algorithm and examine its complexity analysis.

Note that for $k > 1$ we have considerable freedom on the placement of points on segments not incident to longest edges, and therefore a very large number of optimal solutions may exist. To reduce the search space, we

[1] A parameterized problem k-P with input $\langle x, k\rangle$ is *fixed-parameter tractable* if solvable in time $f(k)\text{poly}(|x|)$. Here, $f(.)$ is an arbitrary function (depending on parameter k only) and poly(.) is a polynomial function in the input size [6].

[2] To show fixed-parameter tractability simply determine for each of the 2^k point sets in polynomial a best α and then return the maximum α of all these.

constrain the solution to be determined. In order to compare different (spanning tree) solutions we define the following. For any two selections of points on segments, and for their corresponding spanning trees, let L and L' be ordered list of lengths of all edges in the two trees, sorted from longest to shortest. That is, $L = (l_1, l_2, \ldots, l_{n+k-1})$ and $L' = (l'_1, l'_2, \ldots, l'_{n+k-1})$, with $l_i \geq l_{i+1}$ and $l'_i \geq l'_{i+1}$ for all i. We say that L *dominates* L' if for a certain i, $l_i < l'_i$, and $l_j = l'_j$ for any $j < i$. This defines a general ordering on lists where two lists are incomparable only if they are identical. Our algorithm seeks to choose points on segments and a spanning tree such that the corresponding list of edge lengths dominates *all other possibilities* of point selections on the segments. We call such a tree a *minimum solution tree*. In a minimum solution tree, not only are longest edges as short as possible, but also the number of longest edges is minimum. Further, for all i the i^{th} longest edge is as small as possible, and the number of edges of that length is minimum. A choice of points on segments that results in a minimum solution tree is an *optimal point set*.

The above conditions imply convenient properties on the optimal point set w.r.t. a minimum solution tree. Note that, for any point p of an optimal point set, it is impossible to *improve* the solution by slightly moving p on its segment as this would lengthen one of the longest edges incident to p. We distinguish the possibilities for a point p in an optimal point set (see Figure 2).

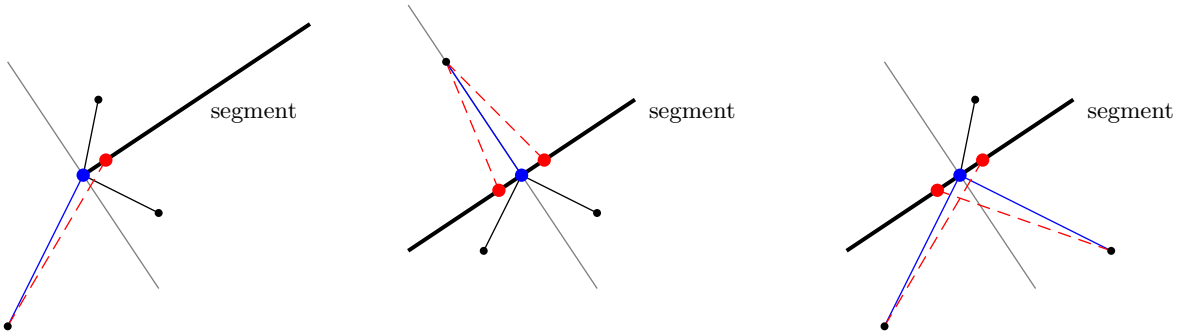

Fig. 2. Points (in blue) of type 1, 2, and 3 respectively, with longest incident edges in blue. In each case, moving the point along the segment results in a longer longest incident edge.

1. Point p lies at an extremity of the segment. Then, one of the longest tree edges incident to p lies on the half plane which is delimited by a line perpendicular to the segment and does not contain the segment. Moving p would lengthen that edge.
2. Point p is on the relative interior of the segment and one of the longest edges of the tree incident to the point is perpendicular to the segment. Moving p in any direction would lengthen that edge.
3. Point p is on the relative interior of the segment and not of type 2. Then, two longest tree edges incident to p lie on different half-planes delimited by a line perpendicular to the segment passing through that point. Moving p in any direction would increase the length of one of these two edges.

For any point p of type 1 or 2, let us call any incident edge *fixing edge* if it would become longer when p was moved, and for any point p of type 3, let us call any pair of incident edges *fixing pair* if one of them would become longer when moving p. Notably, if we know a fixing edge or fixing pair for a point on a segment, then we can deduce the point's position without any knowledge of other incident edges.

Properties of critical paths. Let $(E_1, \ldots, E_m)$ be a given sequence of fixed points and segments, where E_1 and E_m are fixed points or segments, and $E_2, \ldots, E_{m-1}$ are segments. In the following, we study the possible choices of points p_i on segments E_i such that the tree edges e_i from p_i to p_{i+1} satisfy certain properties.

We next define a for our algorithm important notion, the *critical path*. A critical path consists of points p_i located at selected positions on the segments E_i and are connected by edges e_i such that (1) the edges all have the same length, and (2) no other selection of point locations on these segments results in a sequence where edges are not longer but some are strictly shorter.

Our algorithm essentially consists in a successive computation of critical paths. Problem k-$B^L\alpha$ can be solved this way, because the position of points on segments can be determined just by knowing the critical paths of successive edge lengths in the solution.

We next study the properties of critical paths. Since it is impossible to shorten an edge of a critical path by moving a single point p_i without causing another edge to be longer, all p_i on the critical path must be of one of the three types described above. There is, however, a particular case that needs to be addressed: that of parallel segments. If a consecutive subsequence of elements in $(E_1, \ldots, E_m)$ consists of parallel segments at equal distances, and a path of edges of equal length is defined using point locations on these segments which are collinear and perpendicular to the segments, then none of these points can be moved individually without lengthening some edge. Yet it may be possible to move all points of the subset at once without creating any longer edges (Figure 3). We present the rules governing such cases. These rules are summarized by the following

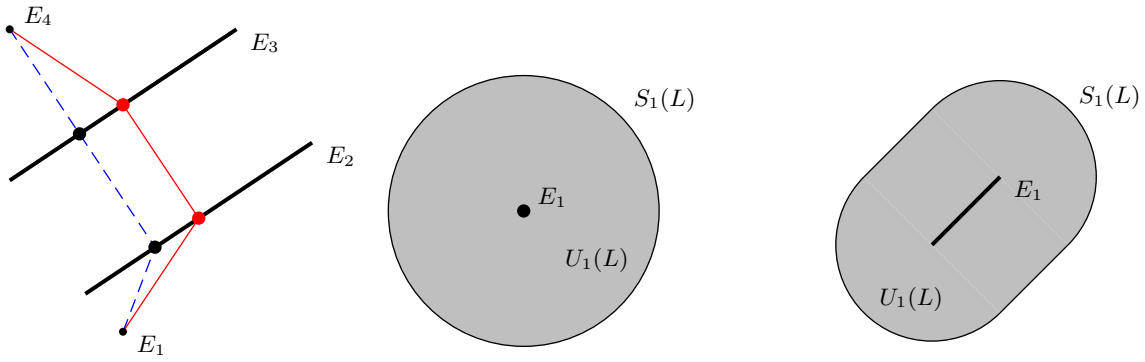

Fig. 3. Left: Choosing points (in red) on E_2 and E_3 results in a path of same-length edges. As they are of type 2, neither can be moved without lengthening the bridging edge; yet they can be both moved simultaneously in a way that shortens some edges of the path without lengthening any other. Examples of $U_1(L)$ (middle and right) and $S_1(L)$ for the cases that E_1 is a fixed point or a segment respectively.

idea: we consider any consecutive subset of the sequence of parallel segments at equal distances as a single segment with the same orientation, equivalent to the intersection of the orthogonal projection of all the parallel segments on a line with the same orientation. Assuming we have a critical path over a sequence $(E_1, \ldots, E_m)$ such that the elements of a consecutive subset $(E_i, \ldots, E_j)$ of segments are all parallel, and the points $p_i, \ldots, p_j$ are the intersection of $(E_i, \ldots, E_j)$ with a single perpendicular line P. Then the following possibilities can occur (Figure 16 in Appendix).

I. Line P intersects with a segment E' at its extremity, $E' \in (E_i, \ldots, E_j)$. Then, either e_{i-1} or e_j lies in the half-plane delimited by P that does not contain E'. This is equivalent with having a point of type 1.

II. Edges e_{i-1} and e_j lie on different half-planes delimited by P. This is equivalent with having a point of type 3.

III. Either edge e_{i-1} or e_j lies on P (but the connected element, E_{i-1} or E_{j+1}, is not a segment parallel to $E_i, \ldots, E_j$). This is equivalent with having a point of type 2.

IV. Line P goes through the extremity of two segments E', E'' in $(E_i, \ldots, E_j)$, such that E' and E'' are in different half-planes delimited by P. Thus, there is a unique line perpendicular to the parallel segments that intersects all segments. There is therefore a unique choice of points connected by edges of the same length, without further conditions required. This is equivalent to considering the sequence of parallel segments as a fixed point.

V. All elements in $(E_1, \ldots, E_m)$ consist of parallel segments at equal distances, and any set of points $p_1, \ldots, p_m$ defined by their intersection with a perpendicular line P defines a critical path. This is equivalent with having a single segment of length equivalent to the intersection of the orthogonal projections of the segments $(E_1, \ldots, E_m)$ on a line with the same orientation.

We show in the following that, apart from sequences of type V, $(E_1, \ldots, E_m)$ supports at most one critical path, i.e. there is a unique choice of point locations on the segments of the sequence that forms a critical path. We call the sequences of type V *degenerate*. We need a few more definitions.

Given $(E_1, \ldots, E_m)$, let $U_1(L)$ be the set of points in the plane reachable from E_1 by an edge of length L or less, and let $U_i(L)$ for $i > 1$ be the set of points on the plane reachable from $U_{i-1}(L) \cap E_i$ by an edge of length L or less. Let further $S_1(L)$ be the set of points on the plane reachable from E_1 by an edge e_1 of length L *exactly*. That is in the case that E_1 is a segment, no point in $S_1(L)$ can be reached from E_1 by an edge a shorter than L. Similarly, let $S_i(L)$ for $i > 1$ be the set of points on the plane reachable from $S_{i-1}(L) \cap E_i$ by an edge e_i of length exactly L, such that $e_1, \ldots, e_i$ form a critical path (assuming the endpoint of e_i that is not on E_i is a fixed point).

We study the properties of $U_i(L)$ and $S_i(L)$. First, we can deduce inductively that if $U_{i-1}(L) \cap E_i \neq \emptyset$, then it consists of a single point or a subsegment (a connected subset) of E_i. If the set $U_i(L) \neq \emptyset$, then $U_i(L)$ is either a ball of radius L, or the Minkowski sum of a ball of radius L and a segment. That is, $U_i(L) = E_1$ if $i = 1$, and $U_i(L) = U_{i-1} \cap E_i$ otherwise (Figure 3).

Lemma 5. $S_i(L) \subseteq U_i(L)$.

Proof. By definition, $S_1(L)$ is a subset of $U_1(L)$. We prove by induction that $S_i(L)$ is a subset of $U_i(L)$ for all i. We remark that it is sufficient to prove that any open set containing a point of $S_i(L)$ intersects with the boundary of $U_i(L)$. By contradiction, let us assume that a point p of $S_i(L)$ belongs to an open set $U \subseteq U_i(L)$. We can then choose two points $q, q' \in U$ such that p is the midpoint of segment qq'. Since $q \in U_i(L)$, there is a set of points $q_1, \ldots, q_i$ in $E_1, \ldots, E_i$ such that the $q_1, \ldots, q_i, q$ are connected by edges of length at most L, with some of them shorter. We define $q'_1, \ldots, q'_i$ for q' analogously. For each pair q_i, q'_i, we define a continuous linear transformation $q_i(\lambda) = (1-\lambda)q_i + \lambda q'_i$, with $0 \leq \lambda \leq 1$. We also define $q(\lambda) = (1-\lambda)q + \lambda q'$, so that $q(0.5) = p$. Let $l_i^2(\lambda)$ represent the square of the length of the edge connecting $q_i(\lambda)$ and $q'_i(\lambda)$. We prove that $l_i^2(\lambda)$ is a convex function. If the coordinates of q_i and q'_i are (q_x, q_y) and (q'_x, q'_y) respectively, then $l_i^2(\lambda) = (q_x + \lambda(q'_x - q_x))^2 + (q_y + \lambda(q'_y - q_y))^2$. Then the second derivative is $2(q'_x - q_x)^2 + 2(q'_y - q_y)^2 \geq 0$, and so $l_i^2(\lambda)$ is convex. So if $l_i^2(0) \leq L^2$ and $l_i^2(1) \leq L^2$, then $l_i^2(\lambda) \leq L^2$, and if one of the former inequalities is strict, then the latter also is. Therefore, $q_1(0.5), \ldots, q_i(0.5), q(0.5) = p$ defines a list of edges of length no more than L, and since some of the edges connecting $q_1, \ldots, q_i, q$ are shorter than L, some of the edges in the path leading to p are shorter than L. This

contradicts the assumption that there is a critical path leading to p with edges all of length L, which implies that there is no other path with shorter edges.

It follows that if $U_{i-1}(L) \cap E_i = \emptyset$, then $S_{i-1}(L) \cap E_i = \emptyset$. If $U_{i-1}(L) \cap E_i$ consists of a single point p, then either $S_{i-1}(L) \cap E_i = \emptyset$ or $S_{i-1}(L) \cap E_i = \{p\}$. If $U_{i-1}(L) \cap E_i$ is a subsegment of E_i, then $S_{i-1}(L) \cap E_i$ can be empty, one extremity of the subsegment, both extremities of the subsegment, or the complete subsegment. In fact, we can prove the following lemma.

Lemma 6. *(a) The set $S_1(L)$ is the boundary of $U_1(L)$. (b) For all $i > 1$, the set $S_i(L)$ is the intersection of the boundary of $U_i(L)$ with the Minkowski sum of a circle of radius L and $S_{i-1} \cap E_i$.*

Proof. (a) follows from the definition of $S_1(L)$. (b) By definition, $S_i(L)$ contains only points at distance L of $S_{i-1} \cap E_i$. From Lemma 5, we know that $S_i(L) \subseteq U_i(L)$. It is therefore sufficient to prove that every point in the intersection is in $S_i(L)$. We prove by induction. Let p be any point in the intersection. Since p is in the Minkowski sum of a circle of radius L and $S_{i-1} \cap E_i$, there exists $q \in S_{i-1} \cap E_i$ at distance exactly L of p, and by induction hypthesis, there is a path of edges of length L to q, which we can extend to p. We need to prove that there is no other path to p that uses edges no longer than L, and some shorter. Suppose there is a choice of $p_1, \ldots, p_i$ such that $p_1, \ldots, p_i, p$ is such a path. Suppose first that the last edge is shorter than L by some $\varepsilon > 0$. Then, by changing the length of the last edge by less than ε, we can find paths to any point in some open set around p. This contradicts the assumption that p is part of the boundary of $U_i(L)$. Therefore, the last edge of the path is of length L exactly. But that means that p is at distance L of both q and p_i, which are both in $U_i(L) \cap E_i$. This means that the midpoint of the segment from q to p_i is in $U_i(L) \cap E_i$ and at distance less than L from p, yielding a contradiction.

The following cases are possible (Figure 4).

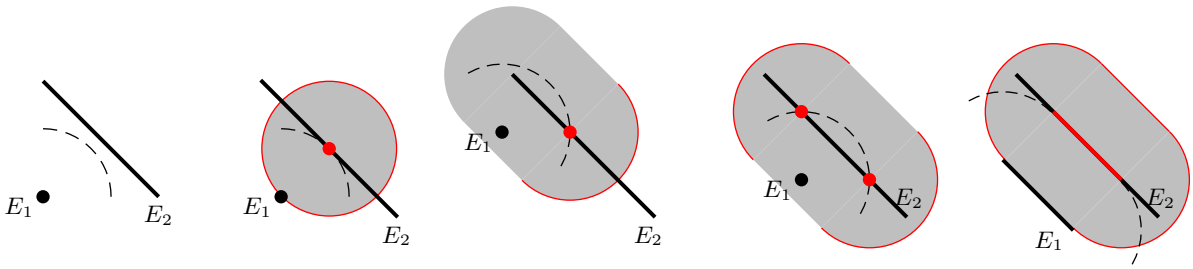

Fig. 4. Shapes of $S_i(L)$ of type a, b, c, d and e respectively. $S_1(L)$ is indicated with a dashed line, $S_1(L) \cap E_2$ and $S_2(L)$ are in red.

a. $S_{i-1}(L) \cap E_i = \emptyset$ and therefore $S_i(L) = \emptyset$.
b. $U_{i-1}(L) \cap E_i$ consists of a single point p and $S_{i-1}(L) \cap E_i = \{p\}$. Then $S_i(L)$ is a circle of radius L centered around p.
c. $U_{i-1}(L) \cap E_i$ is a subsegment of E_i and $S_{i-1}(L) \cap E_i$ is a single extremity of the subsegment. In this case, U_i is the Minkowski sum of the subsegment and a ball of radius L, and S_i is the half circle of radius L centered on $S_{i-1}(L) \cap E_i$ at the "one end" of U_i.

d. $U_{i-1}(L) \cap E_i$ is a subsegment of E_i and $S_{i-1}(L) \cap E_i$ consists of both extremities of the subsegment. In this case, U_i is the Minkowski sum of the subsegment and a ball of radius L, and S_i consists of both half circles of radius L each centered on a point of $S_{i-1}(L) \cap E_i$ at both "ends" of U_i.
e. $U_{i-1}(L) \cap E_i$ is a subsegment of E_i and $S_{i-1}(L) \cap E_i$ is the same subsegment. In this case, U_i is the Minkowski sum of the subsegment and a ball of radius L, and S_i is the boundary of U_i.

For a given $(E_1, \ldots, E_m)$ and length L it is therefore possible to compute successively the $U_i(L)$'s and $S_i(L)$'s. In order to obtain a critical path we require $U_{m-1}(L) \cap E_m = S_{m-1}(L) \cap E_m$. Notably, his can only happen for the smallest L such that $U_{m-1}(L) \cap E_m \neq \emptyset$.

Given $(E_1, \ldots, E_m)$ and L, the following algorithm determines an existing critical path with the required precision. (1) Determine $U_{m-1}(L)$. (2) If $U_{m-1}(L) \cap E_m = \emptyset$, then increase L and go to (1). (3) If $U_{m-1}(L) \cap E_m \neq \emptyset$, then decrease L and go to (1). Once the approximate minimum L^* is found, check whether $U_{m-1}(L^*) \cap E_m = S_{m-1}(L^*) \cap E_m$. Possible outcomes are for $U_{m-1}(L) \cap E_m \neq \emptyset$ (Figure 17 in Appendix):

α. For the minimum L^* where E_m intersects the interior of $U_{m-1}(L^*)$, there is no critical path.
β. For the minimum L^* with $U_{m-1}(L^*) \cap E_m = S_{m-1}(L^*) \cap E_m = \{p\}$, there is a critical path.
γ. For the minimum L^* with $U_{m-1}(L^*) \cap E_m$ is a single point and $S_{m-1}(L^*) \cap E_m = \emptyset$, there is no critical path.
δ. For the minimum L^* with $U_{m-1}(L)^* \cap E_m = S_{m-1}(L^*) \cap E_m$ is a segment, the sequence is a degenerate sequence of parallel segments, and there are infinitely many critical paths.
ϵ. For the minimum L^* with $U_{m-1}(L) \cap E_m$ is a segment and $S_{m-1}(L) \cap E_m$ consists only of extremities of the segment, there is no critical path.

Description of the algorithm. In this subsection we show how to compute an optimal solution by examining possible critical paths.

Lemma 7. *If $(E_1, \ldots, E_m)$ supports a critical path of longest edges in an optimal solution, and if there is a unique choice of point locations that defines the critical path, then this choice is part of the optimal solution.*

Note that if we have an oracle giving us a $(E_1, \ldots E_m)$ that supports a critical path of longest edges in the optimal solution, then we can determine the choice of points on these elements in the optimal solution. We can then replace the segments in the sequence by fixed points and solve the rest of the problem separately. This eventually allows us to replace all segments by fixed points, and solve the problem using an algorithm provided in the literature [5]. Note that while the longest edge in an optimal solution might connect fixed points, in the remainder of this section we are only interested in longest edges incident to points on segments.

Lacking an oracle we determine these sequences *by complete enumeration* of *all possible sequences* $(E_1, \ldots, E_m)$ with E_1 and E_m are fixed points or segments, and $E_2, \ldots, E_{m-1}$ segments. This enumeration accounts for most of the complexity of our algorithm. For each sequence in the enumeration, we check whether it supports a critical path and and whether the edge-length for this path is best. If not, we discard this path. That is, we prune these sequences in the enumeration as we find them. Once gone through the complete list possible sequences, we have found the critical path with shortest longest edges. and replace the segments of the sequence with fixed points defined by the critical path. We then execute the algorithm recursively on the thus reduced instance, using edges of length no more than the ones in the critical path just found.

Once the instance does not contain any more segments, we connect all remaining connected components in polynomial time using an algorithm for a given point set.

```
Procedure Solve(instance,max)
  if instance contains a segment then
```

```
    (* find critical path of instance *)
        C* = ∅
        for each (E_1, ..., E_m) with E_1, E_m point or segment and E_2, ..., E_{m-1} segments do
            if (E_1, ..., E_m) is a candidate for a critical path C of value L_C and L_C < max then
                fix points on segments in C
                Solve(modified_instance, L_C)
else
    solve problem for point set
    if solution better than previous one considered then
        save new solution for instance
```

It is crucial to determine critical paths in with progressively decreasing edge lengths since positions of points on segments should only be determined using the longest incident edges.

In the algorithm above we ignored to deal with parallel segments. We need to deal separately with such degenerate sequences consisting of parallel segments only, because they do not determine unique positions for the points on the segments of the sequence. However, if one point on a segment in such a sequence is fixed, all other points are fixed on the same line perpendicular to all segments. Therefore, after finding such an undetermined critical path, we explore the problem recursively as usual, trying to find critical paths with shorter edges. Whenever the algorithm fixes one of the points on the degenerate sequence, all other points are fixed at the same time.

Notably, for any sequence that contains one or two fixed points, the critical path fixes at least one point on a segment.Thus, during the enumeration, we explore in the worst case all possible ways to cover with paths a forest defined on k segments and $2k$ fixed points. The enumeration is exponential in the number k of segments, which is not surprising since we have shown the problem without fixed points is NP-hard. For constant k, the problem is however polynomial in the number n of segments, as our running time analysis will show.

Consider the (multiple recursive) enumerations done by the algorithm. At each enumeration step the algorithm enumerates all possible candidates for critical paths. Since, a critical path contains at most two fixed points, one at each extremity, the enumeration considers only $O(n^2)$ paths for n fixed points. Since every recursive enumeration is performed on a problem that has at least one less segment that the previous one, the recursion tree has a depth of at most k. Therefore, the algorithm explores at most $O(n^{2k})$ paths w.r.t. the number n of fixed points and k segments in the original instance. Once all critical paths have been found, the algorithm uses the result in [5] for fixed points which runs in $O(n\alpha(n))$, the total complexity of the algorithm is in $O(n^{2k+1}\alpha(n))$.

5 Future Work

This work leads to many interesting algorithmic and complexity questions that are exciting and worth investigating further.

We have shown NP-hardness results for the $B\alpha$ problem if the uncertain regions are represented as point pairs, line segments and squares. It will be nice to prove a similar result for other regions like disks. Is it possible to use techniques from convex optimization to design approximation algorithms for the $B\alpha$ problem for, say, line segments or squares? From the perspective of fixed parameter tractability, we observed that the $B\alpha$ problem is in FPT when the instance consists of $n - k$ fixed points and k pairs of points; the parameter is k. However, we conjecture that the $B\alpha$ problem for the case of line segments is W[1]-hard. Our exact algorithm presented, while

polynomial for fixed k, is exponential in the number k of segments. We suspect that, despite the conjectured parametrized intractability, the running time of the enumeration step in our exact algorithm can be greatly reduced by using a Voronoi diagram.

Although we have been able to obtain several NP-hardness results for the $B\alpha$ Problem, we do not have any for the $W\alpha$ Problem which, a priori, seems harder.

In the search for constant factor approximations, it would be nice to narrow the $(\sqrt{2}, 2)$ gap established in lemmas 3 and 4 for the multiplicative constant on the broadcast-from-center heuristic. Is it possible to fruitfully investigate this problem experimentally? If we assume centers to be picked, say, uniformly at random in a square, and points within the disks to again be picked uniformly at random, can we say anything probabilistically about the average case? We have shown that the "cinch-up" heuristic is a constant factor approximation for the $B\alpha$ Problem for non-overlapping disks. As the number of disks, n, approaches infinity we have a family of examples showing that it can be as bad as a 3/2-approximation. Is there a heuristic that multiplicatively approaches OPT as $n \to \infty$, or at least does better than "cinch-up"?

References

1. H. Alt, E. Arkin, H. Bronnimann, J. Erickson, S. Fekete, C. Knauer, J. Lenchner, J. Mitchell, and K. Whittlesey. Minimum-cost coverage of point sets by disks. *Proc. 22nd ACM Symp. Comp. Geom. (SoCG)*, pages 449–458, 2006.
2. E. M. Arkin and R. Hassin. Approximation algorithms for the geometric covering salesman problem. *Disc. Appl. Mathematics*, 55(3):197–218, 1994.
3. A. E. F. Clementi, P. Penna, and R. Silvestri. On the power assignment problem in radio networks. Technical Report TR00-054, Electronic Colloquium on Computational Complexity, 2000.
4. M. de Berg, J. Gudmundsson, M. J. Katz, C. Levcopoulos, M. H. Overmars, and A. F. van der Stappen. TSP with neighborhoods of varying size. *J. Algorithms*, 57(1):22–36, 2005.
5. C. Delfinado and H. Edelsbrunner. An incremental algorithm for Betti numbers of simplicial complexes on the 3-sphere. *Computer Aided Geometric Design*, 12(7):771–784, 1995.
6. R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer-Verlag, New York, 1999.
7. P. Duchet, Y. O. Hamidoune, M. L. Vergnas, and H. Meyniel. Representing a planar graph by vertical lines joining different levels. *Disc. Mathematics*, 46(3):319–321, 1983.
8. A. Dumitrescu and J. S. B. Mitchell. Approximation algorithms for TSP with neighborhoods in the plane. In *Proc. 12th ACM-SIAM Symp. on Disc. Algorithms (SODA)*, pages 38–46, 2001.
9. B. Fuchs. On the hardness of range assignment problems. In *Proc. 6th Italian Conf. Alg. and Compl. (CIAC)*, pages 127–138, 2006.
10. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman, 1979.
11. J. Gudmundsson and C. Levcopoulos. A fast approximation algorithm for TSP with neighborhoods. *Nord. J. Comput.*, 6(4):469–, 1999.
12. N. Lev-Tov and D. Peleg. Exact algorithms and approximation schemes for base station placement problems. In *Proc. 8th Scand. Workshop Alg. Theo. (SWAT)*, pages 90–99, 2002.
13. N. Lev-Tov and D. Peleg. Polynomial time approximation schemes for base station coverage with minimum total radii. *Computer Networks*, 47(4):489–501, 2005.
14. D. Lichtenstein. Planar formulae and their uses. *SIAM J. Comput.*, 11(2):329–343, 1982.
15. J. S. B. Mitchell. A PTAS for TSP with neighborhoods among fat regions in the plane. In *Proc. 18th ACM-SIAM Symp. on Disc. Algorithms (SODA)*, pages 11–18, 2007.
16. G. Parker and R. L. Rardin. Guaranteed performance heuristics for the bottleneck traveling salesman problem. *Operations Research Letters*, 2(6):269–272, 1984.
17. P. Rosenstiehl and R. E. Tarjan. Rectilinear planar layouts and bipolar orientations of planar graphs. *Disc. Comp. Geom.*, 1:343–353, 1986.
18. Y. Yang, M. Lin, J. Xu, and Y. Xie. Minimum spanning tree with neighborhoods. In *Proc. 3rd Conf. Alg. Aspects on Inf. and Management (AAIM)*, pages 306–316, Berlin, Heidelberg, 2007. Springer-Verlag.

6 Appendix

6.1 NP-hardness of $B\alpha$ with line segments and point pairs

We consider the $B\alpha$ problem for the simple case of non-overlapping uncertainty regions of vertically aligned pairs of points, unit distance apart with integer coordinates. We study the decision version of $B\alpha$ problem for $\alpha = 1$, *i.e.*, we want to decide if G_1 is connected for some choice of points, one for each uncertainty pair. By using a reduction from Planar 3-SAT, we will show that this problem in NP-complete.

Later, we will observe that the same reduction also implies an NP-completeness result for the case of line segments. Moreover, we show that our NP-completeness result for the case of point pairs is, in fact, a gap-producing reduction. This will imply that, unless $P = NP$, there is no polynomial time algorithm with approximation ration smaller than $\sqrt{5}/2$ for this problem.

Overview of the Reduction. In our proof, we show a reduction from Planar 3-SAT, a 3-SAT problem with the added condition that the input formula can be represented as a planar graph. Formally, Planar 3-SAT is defined as follows. Let $\Phi = (X, C)$ be an instance of 3-SAT, with variable set $X = \{x_1, \ldots, x_n\}$ and clauses $C = \{c_1, \ldots, c_m\}$. Each clause consists of exactly three literals, each a variable or its negation. For such an instance, we define a formula graph $H(\Phi)$ as follows: $H(\Phi) = (V, E)$ with vertex set $V = X \cup C$ and edge set $E = E_1 \cup E_2$, such that $E_1 = \{(x_i, x_{i+1}) | i < n\}$, and $E_2 = \{(x_i, c_j) | c_j \text{ contains } x_i \text{ or } \overline{x_i}\}$. A planar 3-SAT instance is one whose corresponding formula graph $H(\Phi)$ is planar. The Planar 3-SAT problem is to determine whether a planar 3-SAT instance Φ is satisfiable. Planar 3-SAT problem is known to be NP-complete [14].

We make use of the fact that, given a planar 3-SAT instance Φ with formula graph $H(\Phi)$, this graph has a planar layout on a $O(n) \times O(n)$ grid [7, 17]. Further, in this layout, the vertices (variables and clauses) can be drawn as horizontal line segments and edges as vertical line segments.

To reduce from Planar 3-SAT to an instance of $B\alpha$, where the uncertainty regions are pairs of points, we make use of a number of gadgets. Specifically, given a layout of a Planar 3-SAT instance using line segments as described above, we replace each horizontal line segment corresponding to a variable by a variable gadget, each horizontal line segment corresponding to a clause by a clause gadget, and each edge by an appropriate vertical sequence of uncertainty pairs. We will argue that there exists a choice of points in each of these uncertainty pairs such that the connectivity graph for $\alpha = 1$, G_1, is connected if and only if the corresponding Planar 3-SAT instance is satisfiable. We next describe the different gadgets in detail and how they may be connected.

The clause gadget. Figure 6 gives a schema that describes the functioning of a clause gadget. Each gate in the schema represents the entry of a connection to a literal, with an open gate representing a contribution of "true". If all gates are closed, then, as suggested by the schema, it is possible that the connectivity graph of the clause gadget is connected, but it will be isolated from the rest of the graph. Also, as the schema suggests, if gates to two literals x_i and x_j are both open, then connections are created between the clause gadget and the gadgets for the literals, but no connection via the clause gadget is made between the literals.

The general form for clause gadgets allows adaptation to individual situations (e.g., the length of the horizontal line segment representing a particular clause gadget in the representation of $H(\Phi)$ by line segments; how many of the horizontal segments representing literals contained in the clause lie below the segment representing the clause and how many above). Figure 5 shows an example of a clause gadget where, in the grid representation of $H(\Phi)$, the clause was represented by a horizontal segment connected to one horizontal variable segment lying above, and two variable segments lying below, the segment for the clause. The dimensions of the clause gadget can be adjusted by adding more uncertainty pairs to the sequences between the connections to the variable gadgets, and

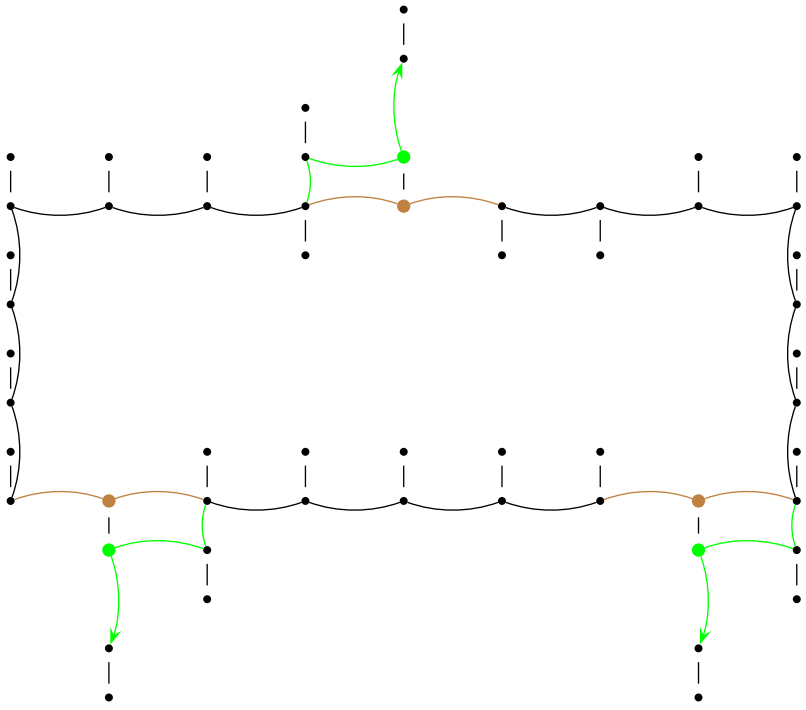

Fig. 5. An example clause gadget. As shown, aside from vertical sequences of uncertainty pairs leading to variable gadgets, there are no other uncertainty pairs in the vicinity of the clause gadget. The graph G_1 can be connected only if at least one of the green points is chosen. The choice of a green point is intended to mean that the attached literal contributes a "true" to the clause.

likewise for distances between connections and distances between connections and the left and right sides of the gadget. As a technical matter not indicated by the figure, it is also straightforward to modify the clause gadget to change by one unit the y-coordinate of a connection to some particular variable gadget without moving the entire gadget.

Consider the three uncertainty pairs with brown and green points in Figure 5. If all three of the brown points are chosen, then it is easy to see that, while points can be chosen so that the connectivity subgraph arising from pairs in the clause gadget can be made connected, no such subgraph can be connected to the rest of G_1 for any choice of points in the remaining uncertainty pairs. If a green point is chosen from a brown-green pair, then the brown edges shown incident to the pair do not belong to G_1.

Each of the three brown-green uncertainty pairs is connected to a variable gadget by a sequence of vertical uncertainty pairs. The choice of a green point, shown in the schema as an open gate, is intended to mean that the literal (a variable or its negation) connecting to this open gate contributes a "true" to the clause. Later subsections outline how (in case points can be chosen to make G_1 connected) variable gadgets transmit truth values and how consistency of truth assignments is assured.

The variable gadget. The variable gadget is shown in Figure 7. Let the uncertainty pair at the extreme left of the variable gadget be the *reference pair* for this variable. We will adopt the interpretation that the choice

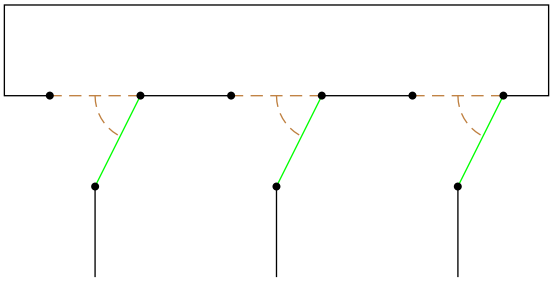

Fig. 6. Schema of the clause gadget. When at least one of the green switches in used, its gate is open and the entire clause gadget can be connected to the rest of the graph.

of blue point in the reference pair means a setting of True to the variable and the choice of red point means a setting of False to the variable.

The variable gadget consists of constructs of type I, II and III. The choice of any blue point in a construct of type I or II forces the choice of blue points in this as well as in all the other constructs of type I and II inside this variable gadget. The same is true for red points. In Figure 7, if the red point is chosen in the reference pair, red arrows show the implications that force the choice of red points in all the type I and II constructs. The function of the type III construct in the variable gadget is to allow this propagation.

Suppose that the variable associated with this gadget is x. If the literal x appears in a clause embedded above the variable gadget, then the connection from the corresponding clause gadget to this variable gadget (to be described in the next section) is made to the top of a construct of type I, *i.e.*, a blue point is chosen in the reference pair. If the literal $\overline{x}$ appears in a clause above the variable gadget, then the connection is made to the top of a construct of type II, *i.e.*, a red point is chosen in a reference pair. Similarly, if the literal x appears in a clause embedded below the variable gadget, the connection from the clause gadget is made to the bottom of a type II construct and the blue point is chosen in the reference pair. If the literal $\overline{x}$ appears in a clause embedded below the variable gadget, the connection is made to the bottom of a type I construct and the red point is chosen in the reference pair.

For any choice of points P from the uncertainty pairs, a necessary and sufficient condition for the subgraph of the connectivity graph G_1 of P restricted to a variable gadget to be connected is that the points in P represent a consistent choice of truth value for this variable. This follows from the observation that a clause gadget cannot connect two different variable gadgets.

We replace horizontal line segment in the embedding of the Planar 3-SAT instance by variable gadgets. Note that the width of the I and II constructs can be adjusted by adding horizontally arranged uncertainty pairs. The number of occurrences of constructs of type I, II and III depend on the number of clauses containing this variable.

Linking the gadgets. We now explain how to represent the edges of the planar graph $H(\Phi)$, corresponding to an instance Φ of Planar 3-SAT. In the embedding we are considering, edges are vertical line segments. They represent two kinds of connections: (1) between a pair of variables, and (2) between a clause and a variable in that clause. Figure 8 shows vertical constructs of uncertainty pairs that connect pairs of variable gadgets and clause and variable gadgets. We observe the following properties of the two connectors: In a clause-variable connector,

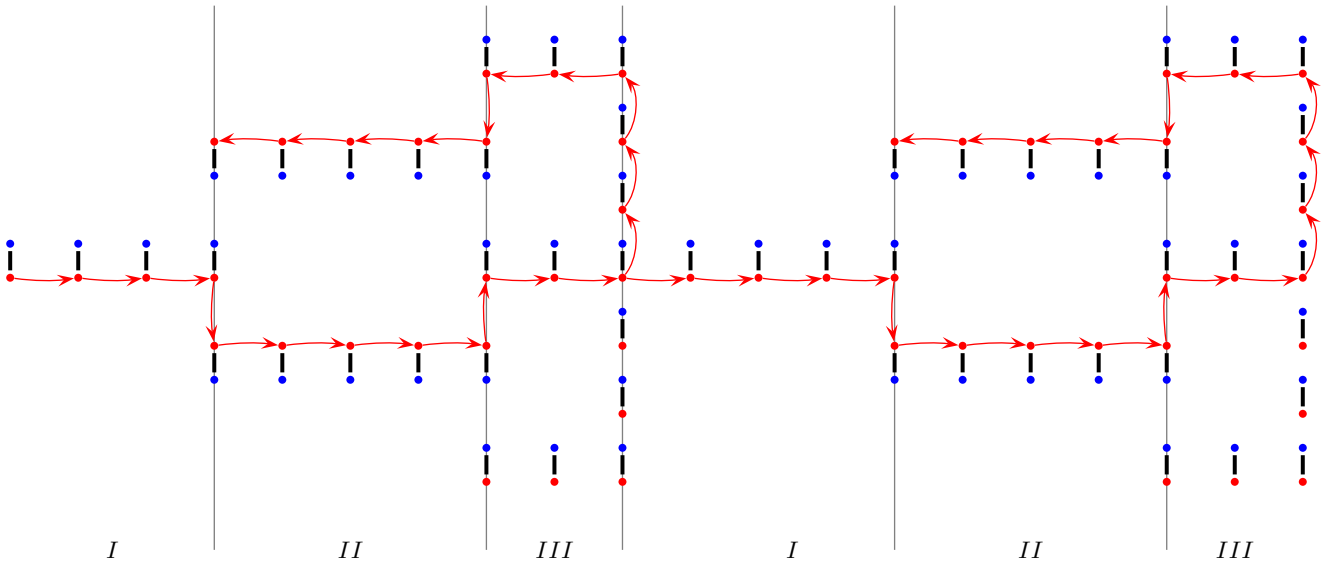

Fig. 7. An example variable gadget.

the choice of blue point in a clause gadget above a variable gadget implies the choice of blue point in the variable gadget. The choice of red point in the clause gadget below a variable gadget forces the choice of red point in the variable gadget. In the variable-variable connector, for any choice of points in the two vertically extreme uncertainty pairs, there is a path using points in the shown uncertainty pairs that connects the two extreme uncertainty pairs. The black uncertainty pair allows this connection. The n variable gadgets are connected using $n-1$ variable-variable connectors.

A minor technical remark: If we need to connect a clause to the top of construct I in one variable gadget and the top of construct II in a different variable gadget, sometimes it is necessary to modify the switches for the two variables in the clause gadget so that one switch is a unit distance above the other. This is possible due to the flexibility inside the clause gadget. We postpone the details to the final version.

The Correctness of the Reduction. In a line segment embedding of planar graph $H(\Phi)$ corresponding to a Planar 3-SAT instance Φ, nodes (clauses and variables) are horizontal line segments and edges are vertical line segments. We have presented clause and variable gadgets to replace horizonal line segments and connectors to replace vertical line segments. We will argue that the connectivity graph G_1 of these uncertainty pairs is connected if and only if the planar 3-SAT instance Φ is satisfiable.

If the planar 3-SAT instance Φ is satisfiable, let us consider the assignment of truth values to the variables of the instance. When a variable is set to True, we choose the blue point in the reference pair of the corresponding variable gadget. When it is set to False, we choose the red point. This will, in turn, lead to the appropriate choice of points in the connectors and inside the clause gadgets. Let P be the set of points chosen and us consider the graph G_1 of P. In G_1, the variable gadgets are all internally connected and connected to each other as a path via the variable-variable connectors. As all the clauses are satisfied by the truth assignment, each clause is connected to one or more variable gadgets through edge-disjoint paths. Therefore the graph G_1 is connected.

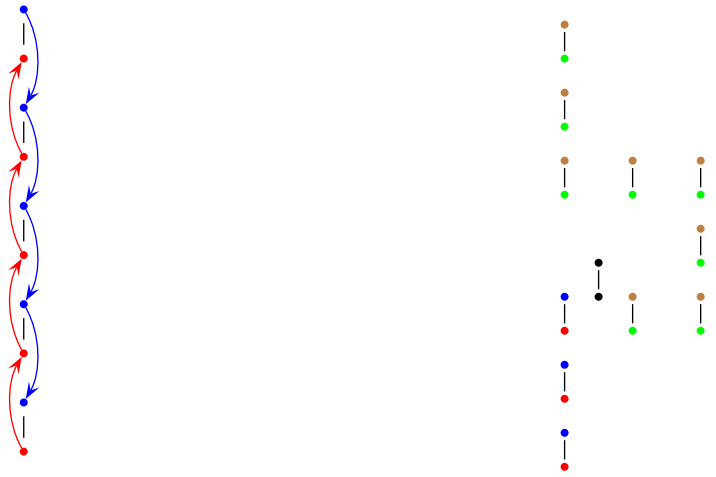

Fig. 8. (Left): A connector between clause and variable gadgets. (Right): A connector between variable gadgets.

Let P be any choice of points for which the corresponding graph G_1 is connected. In G_1, each clause gadget is connected to one or more variable gadgets through edge-disjoint paths. Therefore, two different variable gadgets can never be connected to each other through a clause gadget. This implies that all the variable gadgets have to the internally connected and also connected to each other variable gadgets through a path of length $n-1$. This structure of G_1 gives a a truth assigment for each variable depending on whether the blue or red points are chosen inside the gadget. It is easy to see that this truth assignment satisfies the Planar 3-SAT instance ϕ.

By reduction from Planar 3-SAT, $B\alpha$ for non-overlapping uncertainty regions that are pairs of vertically aligned points one unit apart with integer coordinates is NP-hard.

6.2 Best approximation.

We observe in this section that there is no approximation algorithm, polynomial in the size of the input, with a factor less than $\sqrt{5}/2$, unless $P = NP$. We have proved that there is a Euclidean spanning tree using edges of length at most 2 if and only if there is a valid assignment for the Planar 3-SAT problem. We now examine the case where there is no such assignment. In that case, for any choice of points, there must be an edge in the minimum spanning tree of length more than 2. Since all points have integer coordinates, this edge has a length of at least $\sqrt{5}$.

6.3 Uncertainty on line segments.

If we have uncertainty points on vertical unit segments, we can also prove the best-case α-shape problem is NP-hard, using the same gadgets as presented for pairs and replacing each pair by a vertical unit segment. However, hardness of approximation result does not hold, because we can use edges of length arbitrarily close to 1.

6.4 NP-hardness of the best-case α problem with square uncertainties

As in the previous proof, we reduce the problem Planar-3-SAT to the B_α-Problem with square uncertainties.

Theorem 1. *It is NP-hard to find an exact solution to the B_α-Problem for the case where the regions of uncertainty are squares of edge length one.*

Proof. Our reduction uses a structure similar to the previous proof. We begin with a grid layout of the formula graph corresponding to a Planar 3-SAT instance.

Planar-3-SAT. The planar-3-SAT problem is a 3-SAT problem, with the added condition that if we create a formula graph, variables on one side, clauses on the other side, such that a variable is linked to a clause iff that variable or its negation appears in the clause, then the formula graph is planar. It is known that any planar graph with n vertices can be embedded on a n^2-by-n^2 square grid with edges following the grid.

We need the following terminology.

Terminology. A point p is *l-connected* to a point q if the (Euclidean) distance from p to q does not exceed l. A set S of points is l-connected if the maximum edge length of the minimum spanning tree of S does not exceed l.

The reduction We now describe the variable and the clause gadgets and the connectors we use to link the variables and clauses.

The variable gadget. For any variable, we create the gadget shown in Figure 9.

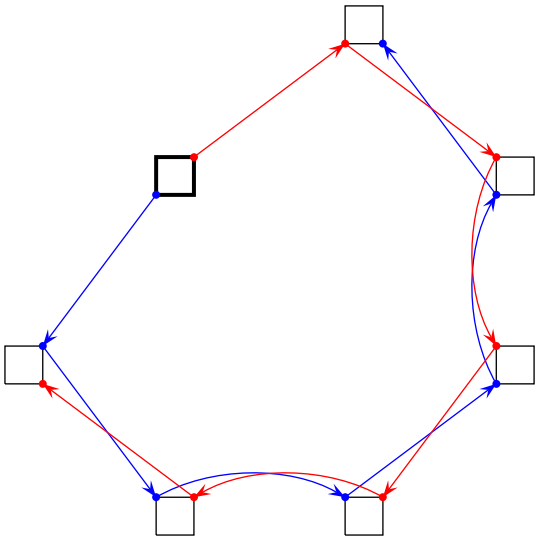

Fig. 9. The variable gadget

In this figure, we have six 'path' squares and a single 'core' square. Each pair of adjacent path squares have coordinates that differ (in x and y) by $(0, 4)$, $(4, 0)$, or $(3, 3)$. The core square is separated by each of its two closest path squares by $(3, 4)$ or $(4, 3)$. The key observation to make is that there are only two possible ways to place a single point within each square such that the points will be 5-connected. In one of these, the core square's point lies at its bottom left corner, so that it is 5-connected (by the Pythagorean theorem: $3^2 + 4^2 = 5^2$) to only one path square's point, as indicated by the solid line. In the other, the core square's point lies at its upper right corner, so that it is 5-connected to a different path square, as indicated by the dotted line. Note that this choice

of core square point propagates along the path squares, forcing the point within each path square to be in one of two corners if they are to be 5-connected to the rest.

The clause gadget. For each clause, we create the gadget of Figure 10. In this gadget, we have a single core

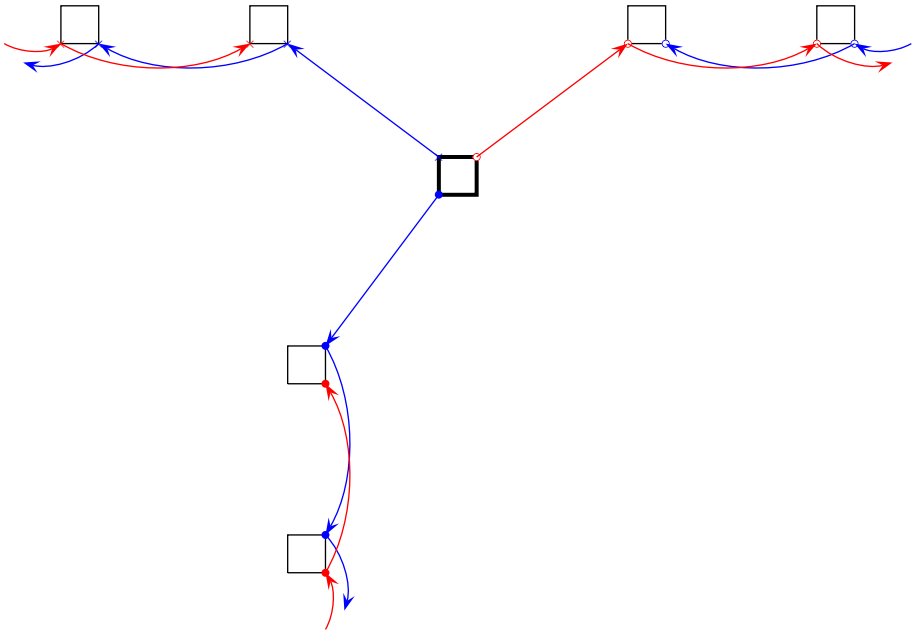

Fig. 10. The clause gadget

square, and three sequences of 'arm' squares. Each arm is labelled with a (possibly negated) literal. Observe that the core square can be 5-connected to only a single arm, and once this arm is selected, its squares' points are *forced*: they must lie in the corners closest to the core square in order for the core point to be 5-connected to the rest of the points (along the solid lines). The points within the other arms' squares are *free*, in the sense that they can move to the corners connected along the dotted lines, since they do not need to connect to the core point.

Linking the gadgets. Variable gadgets are linked to clause gadgets by using a path of squares (Figure 11).

Once a choice is made in the clause gadget of which arm is to connect the core square to the rest, the choice is propagated along the link path to the variable gadget, where the boolean value of the clause arm (positive or negative) must agree with the variable core square's value in order for the clause core square to be 5-connected to the rest. The points within the two arms of the clause that are not 5-connected to the clause core are free to move closer to their respective variable gadgets, so that they will be 5-connected to them even if their boolean values disagree.

A forced point must lie at a particular corner of its square in order to be 5-connected. Figure 11 shows how the choice of corner can be changed, if necessary.

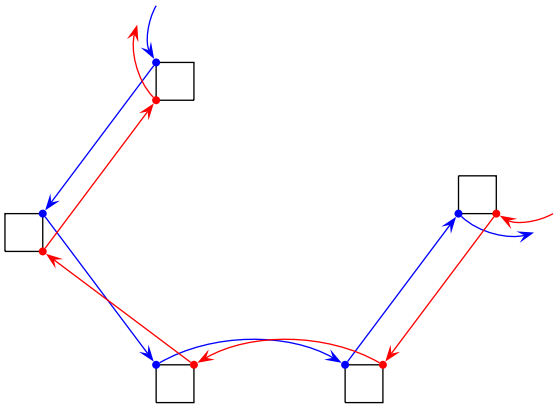

Fig. 11. A path of squares

Sequences can be *split* using the gadget of Figure 12. If the value of a variable gadget connected to the left side of this gadget forces the point within the leftmost square to be at its top left corner, this causes the rest of the squares to be forced (along the solid lines) as shown. Similarly, any free points from the clause gadgets (connected to the top right and bottom right squares) will propagate towards the variable gadget.

Correctness of the Reduction. As a planar-3-SAT instance can be represented as a bipartite planar graph, we do so (this can be done in polynomial time). We then embed this graph on a on a n^2-by-n^2 square grid with edges following the grid (this can be done in polynomial time). We then replace the variable vertices of the graph by a variable gadget, the clause vertices by a clause gadget, and the edges by sequences of squares.

Any variable gadget and its associated sequences of squares can be 5-connected by a single tree, and the central squares of the clause gadgets will be 5-connected to exactly one of those trees iff there is a valid assignment for the planar-3-SAT instance. At this point, there are as many trees as variables. In order to connect this forest of trees into a single 5-connected tree, we add 'loose' connections between the trees by using a sequence of squares which allow the variable gadgets to be 5-connected to each other, regardless of the current values of the variables; see Figure 13.

Finally, if there is an assignment satisfying the planar-3-SAT instance, there is a minimum spanning tree for our construction using edges of length 5 and smaller. Symmetrically, any such tree gives a valid assignment for the planar-3-SAT problem. Hence there is a valid assignment for the planar-3-SAT instance if and only if our construction can be 5-connected in the best case.

6.5 Proof of Lemma 4

We are going to distribute an even number of unit disks with centers equally spaced along a very large (and hence, relative to the unit disks, very flat) circle C. Let us call the distance between consecutive centers of disks centered along the large circle L. We will add more disks, but L will remain the longest edge of a BSpT of the disk centers. Additionally, let us pick $\epsilon \ll L - 2$.

The construction actually contains quite a large number of highly overlapping disks in addition to the disks whose centers lie along C. See figure 14 for an approximate drawing. The drawing is approximate in several respects. First of all, C is much, much larger than drawn, so that if the bottom of C is, say, tangent to the x-axis,

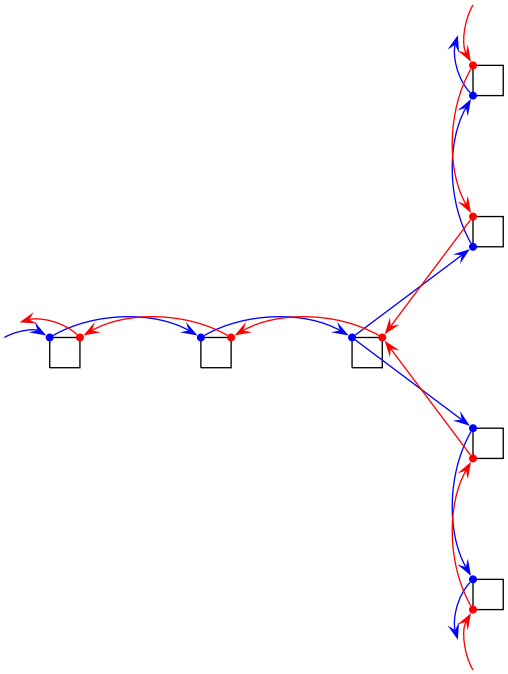

Fig. 12. Splitting the path from a variable to two (or more) clauses

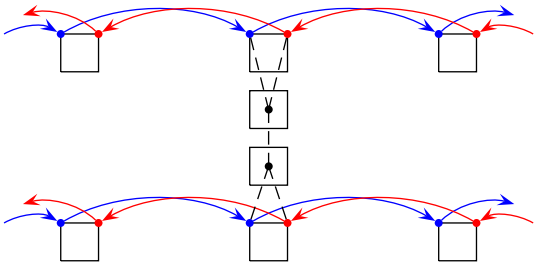

Fig. 13. Connecting variable gadgets

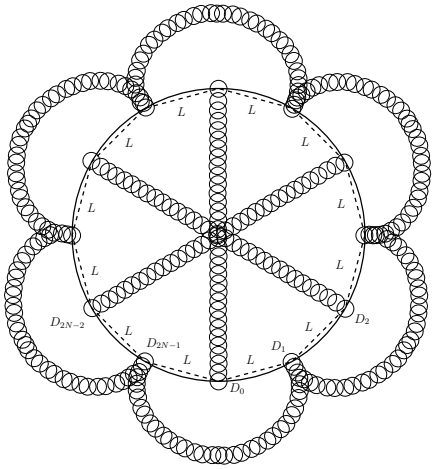

Fig. 14. The construction begins with even number of equally spaced unit disks with centers along a very large circle, C. If the bottom of C is tangent to the x-axis, and the center of the bottom unit disk, D_0, has y-coordinate equal to 0, then the center-points of the first unit disks to the left and right of D_0 along C, each have y-coordinate which is less than $\epsilon/3$. In addition to the disks along C, there is a sequence of disks going from D_0 to its diametrically opposite unit disk with centers lying along the connecting diameter. The center of these disks are all distance ϵ form one another. If we number the disks along C in counter-clockwise order $D_0, ..., D_{2N-1}$, then we have a similar set of diametric disks extending from $D_2, D_4, ..., D_{2N-2}$. The odd numbered unit disks $D_1, D_3, ..., D_{2N-1}$, with representative element that we shall call D_i, each have a set of unit disks running from D_i to D_{i+2} with centers each ϵ from the next, but with the disks running in almost circular patterns on the outside of C.

and the center of the bottom unit disk, D_0 has y-coordinate equal to 0, then the center-points of the first unit disks to the left and right of D_0 along C, each have y-coordinate which is less than $\epsilon/3$. In addition to the disks along C, there is a sequence of disks going from D_0 to its diametrically opposite unit disk whose centers lie along the connecting diameter. The center of these disks are all distance ϵ, one from the next, along the diameter. If we number the C-centered disks in counter-clockwise order, $D_0, ..., D_{2N-1}$, then we have a similar set of diametric disks extending from $D_2, D_4, ..., D_{2N-2}$. The key observation is that we can add such diametrically centered disks in such a way that the center of disks extending from D_j to the center of C, are each more than distance L from the center of any other D_k for $k \neq j$ - thus the choice of D_1 and D_{2N-1} with y-coordinate less than $\epsilon/3$. On the other hand, the odd numbered unit disks $D_1, D_3, ..., D_{2N-1}$, with representative element that we shall call D_i, each have a set of unit disks running from D_i to D_{i+2} with centers each ϵ from the next, but with the disks running in almost circular patterns on the outside of C. An important point in this case, is that the disks start out emanating from D_i along a diametric line, and then bend around so that their centers are never within L of D_{i+1}.

We claim that for such an arrangement of unit disks, the maximum distance between locations $\ell_r \in D_r$ in a BSpT can be as small as (and in fact slightly smaller than) $\sqrt{L^2+4}$, where the set $\{D_r\}$ consists not just of the disks D_i with centers along C, but all the other unit disks depicted in Figure 14 as well. If D_i, D_{i+2} are two consecutive disks in the cyclical ordering of C-centered disks with i odd, let $\{D_{i_k}\}$ denote the set of disks running from D_i to D_{i+2} outside of C. Further, if D_j, D_{j+N} are diametrically opposite C-centered disks with j even, let $\{D_{j_k}\}$ denote the set of disks running diametrically between D_j and D_{j+N}. To verify our claim about $\{\ell_i\}$ with maximum BSpT edge length slightly less than $\sqrt{L^2+4}$, pick $\ell_i \in D_i$ for even i to be the point in D_i

closest to the center of C and $\ell_i \in D_i$ for odd i to be the point in D_i furthest from the center of C. See Figure 15. Regardless of the choice of the $\ell_{i_j} \in D_{i_j}$ it is clear that $\bigcup\{D(\ell_i;\alpha)\} \cup \bigcup\{D(\ell_{i_j};\alpha)\}$ is connected if 2α is the

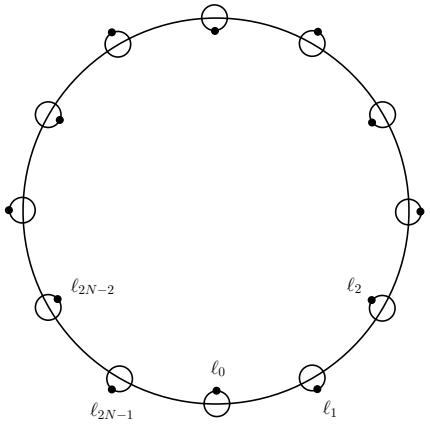

Fig. 15. To verify our claim about $\{\ell_i\}$ with maximum BSpT edge length slightly less than $\sqrt{L^2+4}$, we pick $\ell_i \in D_i$ for even i to be the point in D_i closest to the center of C and $\ell_i \in D_i$ for odd i to be the point in D_i furthest from the center of C. The distance between ℓ_i and ℓ_{i+1} (in the cyclical ordering) is then just slightly less than $\sqrt{L^2+4}$.

distance between consecutive locations ℓ_i, ℓ_{i+1} (in the cyclical ordering), and that this distance is, as claimed, just slightly less than $\sqrt{L^2+4}$. Let us designate this distinguished choice of the $\ell_i \in D_i$ by ℓ_i^*, and the associated α by α^*.

For these $\{D_i\}$ and $\{D_{i_j}\}$, if there were any choice of $\{\ell_i\}, \{\ell_{i_j}\}$ making α any larger, then we would have to pick one of the ℓ_i to the left or right of the diametric line through the center of C and D_i. It is easy to check that the result of such a choice is that there would be some cyclically ordered pair, ℓ_j, ℓ_{j+1} whose distance $d(\ell_j, \ell_{j+1}) < d(\ell_j^*, \ell_{j+1}^*) = \alpha^*$. But then $D(\ell_j;\alpha^*) \cup D(\ell_{j+1};\alpha^*)$ connects ℓ_j, ℓ_{j+1} and $\bigcup\{D(\ell_{2k};\alpha^*)\} \cup \bigcup\{D(\ell_{(2k)_j};\alpha^*)\}$ connects the even-indexed ℓ_{2k} and any associated choices for $\ell_{(2k)_j}$, while $\bigcup\{D(\ell_{2k+1};\alpha^*)\} \cup \bigcup\{D(\ell_{(2k+1)_j};\alpha^*)\}$ connects the odd-indexed ℓ_{2k+1} and any associated choices for $\ell_{(2k+1)_j}$. It follows that $\alpha \leq \alpha^*$, contrary to assumption, and so the fact that OPT can be as small as $\frac{\sqrt{L^2+4}}{2}$ is established. The algorithm of Lemma 2 picked $\alpha = L+1$, so picking L sufficiently close to 2 yields $\frac{\frac{L}{2}+1}{\frac{\sqrt{L^2+4}}{2}} = \frac{L+2}{\sqrt{L^2+4}}$ sufficiently close to $\sqrt{2}$, completing the proof.

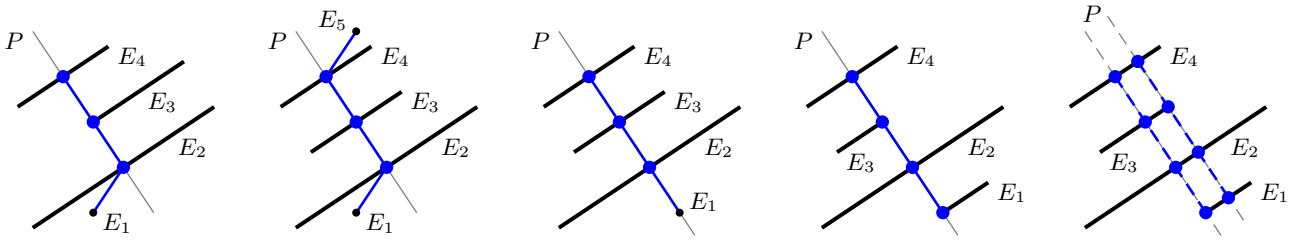

Fig. 16. Sequences of type I, II, III, IV and V respectively. The critical paths are shown in blue. Note that in the case of type V, the critical path can be moved at will in the indicated range. For all other sequences, any different choice of blue points on the segments results in at least one longer edge.

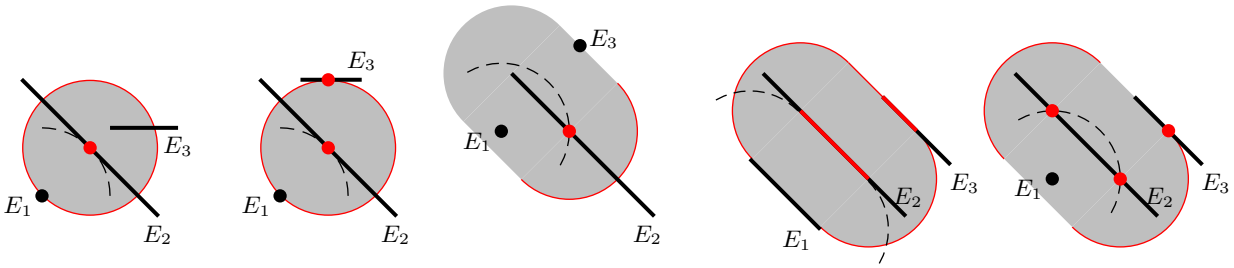

Fig. 17. Sequence (E_1, E_2, E_3) with outcomes of type α, β, γ, δ and ϵ respectively for the case that L is minimum such that $U_2(L) \cap E_3 \neq \emptyset$. $S_1(L)$ is depicted by dashed lines. $S_1(L) \cap E_2$, $S_2(L)$ and $S_2(L) \cap E_3$ are shown in red. Note that in the outcome of type 1, a smaller L would result in $U_1(L) \cap E_2 = \emptyset$, and therefore $U_2(L) = \emptyset$.