Connectivity graphs of uncertainty regions*

Erin Chambers[1], Alejandro Erickson[2], Sándor Fekete[3], Jonathan Lenchner[4], Jeff Sember[5], S. Venkatesh[2], Ulrike Stege[2], Svetlana Stolpner[6], Christophe Weibel[7], and Sue Whitesides[2]

[1] Dept. of Mathematics and Computer Science, Saint Louis University echambe5@slu.edu
[2] Dept. of Computer Science, University of Victoria {ate,sue}@uvic.ca,{venkat,stege}@cs.uvic.ca
[3] Inst. of Operating and Computer Networks, TU Braunschweig s.fekete@tu-bs.de
[4] IBM T.J. Watson Research Center lenchner@us.ibm.com
[5] Dept. of Computer Science, University of British Columbia jpsember@cs.ubc.ca
[6] School of Computer Science, McGill University sveta@cim.mcgill.ca
[7] Computer Science Department, Dartmouth College weibel@cs.dartmouth.edu

Abstract. We study a generalization of the well known bottleneck spanning tree problem called *Best Case Connectivity with Uncertainty*: Given a family of geometric regions, choose one point per region, such that the length of the longest edge in a spanning tree of a disc intersection graph is minimized. We show that this problem is NP-hard even for very simple scenarios such as line segments and squares. We also give exact and approximation algorithms for the case of line segments and unit discs respectively.

1 Introduction

Finding an optimally connected substructure in a network is one of the fundamental combinatorial 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. This corresponds to choosing the necessary power and thus range for the routers to be placed at nodes. Often, greedy methods still yield optimal solutions. However, the situation changes when the location of devices becomes part of the problem: How should each location be chosen from a given neighborhood, such that the solution to the resulting bottleneck connectivity problem is optimal? 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).

* 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.

Let U, $|U| = n$, denote a family of uncertainty regions, e.g., a family of disks, squares, line segments or pairs of points. For each uncertainty region $u_i \in U$, $1 \leq i \leq n$, one point p_i is to be chosen inside this region u_i. 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 $E = \{(p_i \in P, p_j \in P), \|p_i - p_j\|_2 \leq 2\alpha\}$. Thus, the graph connects a pair of points with an edge whenever closed disks of radius α centered at these points intersect. We can now formally define the main problem, *Best Case Connectivity with Uncertainty* (BCU), that we study in this paper.

The BCU Problem. Given a set $U = \{u_1, \ldots, u_n\}$ of n uncertainty regions, find the minimum value α for which there exists a choice of point set $P = \{p_1, \ldots, p_n\}, p_i \in u_i$, such that the connectivity graph G_α of P is connected.

Related Work. If the n uncertainty regions are points (in other words, there is no uncertainty), then finding the minimum α for which the connectivity graph is connected amounts to finding a minimum Euclidean Bottleneck Spanning Tree (MBST) on the points. Since minimum spanning trees (MSTs) are also MBSTs, these can be found in time $O(n \log n)$.

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, 10, 11, 1, 7]). Thus our work provides an early exploration of connectivity problems, arising in the context of wireless networks, for points lying in nontrivial uncertainty regions.

The minimum spanning tree problem has been studied in the setting of uncertainty regions. Yang *et al.* [16] 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 admit a PTAS, unless P=NP. Other optimization problems with neighborhoods that have received attention include the Traveling Salesman Problem; e.g. see [2, 9, 6, 4, 13]. The bottleneck version of TSP is known to be NP-hard [8, p. 212]. A 2-approximation has been known since 1984 [14].

Our Main Results. After sketching that many variants of BCU are NP-hard (some even to approximate), we give exact and approximation algorithms for certain variants. Given the geometric nature of our problems, we use the Euclidean measure of distance. Our main results are as follows:

1. We show that BCU is NP-hard even in the simple case when the uncertainty regions are vertical line segments. Our proof technique also works when the regions are all squares or even pairs of points; We show that it is NP-hard to approximate BCU within a factor less than $\sqrt{5}/2$ when the uncertainty regions are pairs of points. See Section 2.

2. We present an exact algorithm for BCU when the instance consists of n fixed points and k line segments. The algorithm is polynomial in n for constant k. See Section 3.
3. For uncertainty regions that are all unit disks, we give a simple constant additive approximation algorithm for this problem. A slight modification of this algorithm gives a constant multiplicative approximation in case the disks are *non-overlapping*. See Section 4.

2 Hardness Results

We prove hardness results for two variants of the BCU problem. Our first theorem shows NP-hardness when the uncertainty regions are line segments or point pairs. Interestingly, this result also implies a hardness of approximation result for the case of point pairs. (see Section A.1 in the appendix). Our second theorem proves NP-hardness for the case of non-overlapping square uncertainty regions. In the remainder of this section, we sketch the main ideas behind the first result. Appendix A.1 gives the full proofs of both results.

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

Overview of the Reduction. We use a reduction from Planar 3-SAT, 3-SAT with the added condition that the input formula can be represented as a planar graph. 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 an $O(n) \times O(n)$ grid [5, 15]. 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 BCU, where the uncertainty regions are pairs of points, we design various 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 point 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.

Overview of the Gadgets. We give the main ideas behind the clause gadget, variable gadget and connector gadgets linking others.

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 grid layout with horizontal and vertical segments. 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 must be open. This will correspond to setting the literal to "true" in the clause. This, in turn, ensures that the clause is satisfied. See Figure 6 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 occurrence 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 pairs are the same. If not, the subgraph of G_1 corresponding to this variable gadget is not connected. Furthermore, G_1 itself cannot be connected as it cannot join subgraphs arising from parts of a variable gadget. See Figure 7 in the Appendix (Section A.1).

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 (Section A.1). The Appendix (Section A.1) provides further detail including correctness. We finish this section by noting that the same proof works when the point pairs are replaced by a unit length vertical line segment joining them.

3 An exact algorithm solving BCU for n fixed points and k segments

We present an exact algorithm that solves our problem when the input consist of n fixed points and k line segments of any length and orientation (as uncertainty regions). For the ease of presentation, we assume the line segments to be in general position. Our algorithm determines, in a time that is polynomial in n for any fixed k, 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.

Algorithm Overview. A key feature of our recursive algorithm is that it restricts an exhaustive search for an optimal solution to a search of candidate solutions in the set of what we call *minimum solution trees*, optimal solutions that satisfy additional properties. Our algorithm computes the combinatorial structure of candidates for a minimum solution tree by exhaustive search for possible "support sequences" $(E_1, \ldots, E_m)$ for candidate *critical paths*. Their support sequences implicitly determine the locations of the tree vertices on the

segments of the support sequences. The coordinates are specified up to user-defined precision δ. Once a critical path is found, the original problem is updated with new fixed points, specified to precision δ, and fewer segments. At the end the combinatorial structure of a minimum solution tree candidate is known, together with support sequences for critical paths through tree vertices on segments. The exact locations of these vertices are given implicitly by the support sequences. Whenever a fixed point is determined for each segment, we compute a MBST on these points to obtain the α for this minimum solution tree candidate. The candidate tree with the minimum α gives an optimum solution to our input.

Minimum Solution Trees. To prepare for the description of our algorithm, we start with describing properties of the positions of points on segments in an optimal solution. We remark that in an optimal solution for $k > 1$ we can have considerable freedom on the placement of points on segments not incident to longest edges, and therefore even an infinite number of optimal solutions may exist. To reduce the search space, we constrain the solution to be determined in defining a way to compare different (spanning tree) solutions. For this, consider the set of all spanning trees taken over all fixed points and all point choices on the k segments. We define a linear ordering on this set as follows.

For any two selections of points on the k segments, and for two of their corresponding spanning trees, let $\mathcal{L}$ and $\mathcal{L}'$ be ordered lists of lengths of all edges in the two trees, sorted from longest to shortest. That is, $\mathcal{L} = (l_1, l_2, \ldots, l_{n+k-1})$ and $\mathcal{L}' = (l'_1, l'_2, \ldots, l'_{n+k-1})$, with $l_i \geq l_{i+1}$ and $l'_i \geq l'_{i+1}$ for all i. We say that $\mathcal{L}$ *is preferred over* $\mathcal{L}'$ if for a certain i, $l_i < l'_i$, and $l_j = l'_j$ for all $j < i$. This defines a general ordering on lists. Our algorithm seeks to choose points on segments and a spanning tree such that the corresponding list of edge lengths is preferred over *all other possibilities* of point selections on the segments. We call such a tree a *minimum solution tree* $\mathcal{T}$.[8] In a minimum solution tree $\mathcal{T}$, not only are longest edges as short as possible, but also the number of longest edges is minimum. In other words, the tree with a smallest number of shortest longest edges is preferred over the ones with more edges of the same length. 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 $\mathcal{T}$ is an *optimal point set* for $\mathcal{T}$.

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; in fact, any perturbation of a point must lengthen at least one of the edges that is longest among all edges incident to p. We distinguish the possibilities for a point p on a segment in an optimal point set (see Figure 1). Given a point p on a segment, we call an edge of a minimum solution tree incident to p *locally longest* if no other tree edge incident to p is longer.

[8] We remark that for lists $\mathcal{L}$ and $\mathcal{L}'$ for two different spanning tress with two different sets of points on the segments, it is possible that $\mathcal{L} = \mathcal{L}'$, that is we can have a tie.

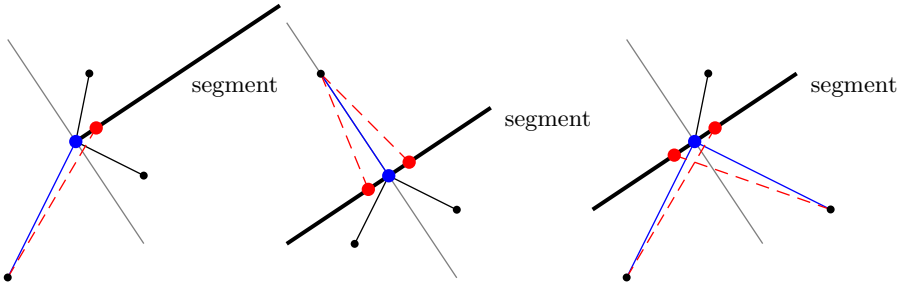

Fig. 1. Points (in blue) of type 1, 2, and 3 respectively, with longest incident edges in blue. In each case, moving the point along the segment results in a longer longest incident edge.

1. Point p lies at an extremity of the segment. Then, one of the locally longest edges incident to p lies on the half plane that is delimited by a line perpendicular to the segment and that 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 locally longest edges incident to p 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 there are two locally longest edges incident to p lying on different half-planes delimited by a line perpendicular to the segment passing through p. Moving p in any direction would increase the length of one of these two edges.

Notably, if we know for any point p on a segment of type 1 or 2 that it has a locally longest incident edge that would become longer if p were moved or if we know for a point p of type 3 that it has a pair of locally longest incident edges where one of them would become longer when moving p, then we can deduce p's position without any knowledge of other incident edges of p.

Critical paths. To define this concept, 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. 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.

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 the p_i on the critical path must be of one of the three types described above.

Before we can describe how to compute a critical path and argue that a sequence $(E_1, \ldots, E_M)$ supports at most one critical path, we need a few more

definitions. Given $(E_1, \ldots, E_m)$ and a positive number Λ, let $U_1(\Lambda)$ be the set of points in the plane reachable from E_1 by an edge of length Λ or less, and let $U_i(\Lambda)$ for $i > 1$ be the set of points on the plane reachable from $U_{i-1}(\Lambda) \cap E_i$ by an edge of length Λ or less. Let $S_1(\Lambda)$ be the set of points on the plane reachable from E_1 by an edge e_1 of length *exactly* Λ, that is in the case that E_1 is a segment, no point in $S_1(\Lambda)$ can be reached from E_1 by an edge shorter than Λ. Similarly, let $S_i(\Lambda)$ for $i > 1$ be the set of points on the plane reachable from $S_{i-1}(\Lambda) \cap E_i$ by an edge e_i of length exactly Λ, 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 $U_i(\Lambda)$ and $S_i(\Lambda)$. First, we can deduce inductively that if $U_{i-1}(\Lambda) \cap E_i \neq \emptyset$, then it consists of a single point or a subsegment (a connected subset) of E_i: If the set $U_i(\Lambda) \neq \emptyset$, then $U_i(\Lambda)$ is either a ball of radius Λ, or the Minkowski sum of a ball of radius Λ and a segment where $U_i(\Lambda) = E_1$ if $i = 1$, and $U_i(\Lambda) = U_{i-1}(\Lambda) \cap E_i$ otherwise (Figure 2).

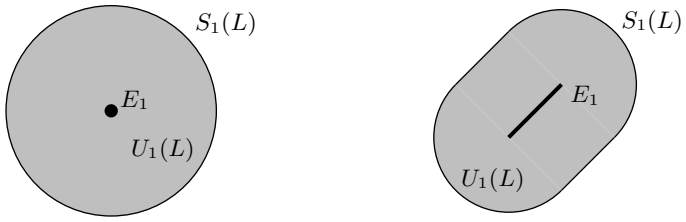

Fig. 2. Examples of $U_1(\Lambda)$ and $S_1(\Lambda)$ for the cases that E_1 is a fixed point or a segment.

Lemma 1. *$S_i(\Lambda) \subseteq U_i(\Lambda)$.*

Proof. Sketch: By definition, $S_1(\Lambda)$ is a subset of $U_1(\Lambda)$. We prove by induction that $S_i(\Lambda)$ is a subset of $U_i(\Lambda)$ for all i. We remark that it is sufficient to prove that any open set containing a point of $S_i(\Lambda)$ intersects with the boundary of $U_i(\Lambda)$, which can be done with some care by contradiction.

It follows that if $U_{i-1}(\Lambda) \cap E_i = \emptyset$, then $S_{i-1}(\Lambda) \cap E_i = \emptyset$. If $U_{i-1}(\Lambda) \cap E_i$ consists of a single point p, then either $S_{i-1}(\Lambda) \cap E_i = \emptyset$ or $S_{i-1}(\Lambda) \cap E_i = \{p\}$. If $U_{i-1}(\Lambda) \cap E_i$ is a subsegment of E_i, then $S_{i-1}(\Lambda) \cap E_i$ 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 2. *(a) The set $S_1(\Lambda)$ is the boundary of $U_1(\Lambda)$. (b) For all $i > 1$, the set $S_i(\Lambda)$ is the intersection of the boundary of $U_i(\Lambda)$ with the Minkowski sum of a circle of radius Λ and $S_{i-1}(\Lambda) \cap E_i$.*

Proof. (a) This follows from the definition of $S_1(\Lambda)$. (b) By definition, $S_i(\Lambda)$ contains only points at distance Λ from $S_{i-1} \cap E_i$. From Lemma 1, we know

that $S_i(\Lambda) \subseteq U_i(\Lambda)$. It is therefore sufficient to prove that every point in the intersection is in $S_i(\Lambda)$. We prove this by induction. Let p be any point in the intersection. Since p is in the Minkowski sum of a circle of radius Λ and $S_{i-1} \cap E_i$, there exists $q \in S_{i-1} \cap E_i$ at distance exactly Λ from p, and by the induction hypothesis, there is a path of edges of length Λ from 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 Λ, 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 Λ 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 $U_i(\Lambda)$. Therefore, the last edge of the path is of length Λ exactly. But that means that p is at distance Λ from both q and p_i, which are both in $U_i(\Lambda) \cap E_i$. This means that the midpoint of the segment from q to p_i is in $U_i(\Lambda) \cap E_i$ and at distance less than Λ from p, yielding a contradiction. □

The following cases are possible (Figure 3).

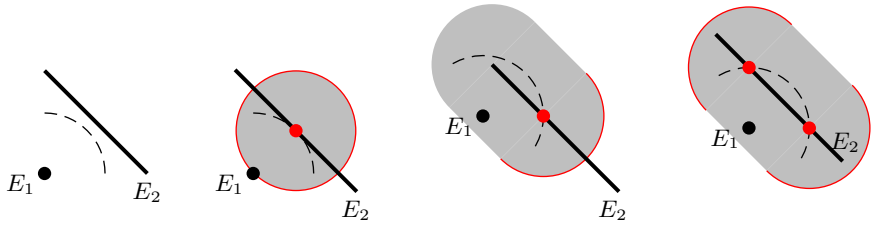

Fig. 3. Shapes of $S_i(\Lambda)$ of type a, b, c and d respectively. $S_1(\Lambda)$ is indicated with a dashed line, $S_1(\Lambda) \cap E_2$ and $S_2(\Lambda)$ are in red.

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

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

In our exact algorithm critical paths contribute to the set of candidate solutions. Given $(E_1, \ldots, E_m)$ and Λ, the following algorithm determines an existing critical path with a precision, say δ.

(1) Determine $U_{m-1}(\Lambda)$. (2) If $U_{m-1}(\Lambda) \cap E_m = \emptyset$, then increase Λ and go to (1). (3) If $U_{m-1}(\Lambda) \cap E_m \neq \emptyset$, then decrease Λ and go to (1). Once the approximate minimum $L^* \in [\Lambda - \delta, \Lambda + \delta]$ is found, check whether $U_{m-1}(L^*) \cap E_m = S_{m-1}(L^*) \cap E_m$.

Possible outcomes for $U_{m-1}(\Lambda) \cap E_m \neq \emptyset$ (Figure 4) are:

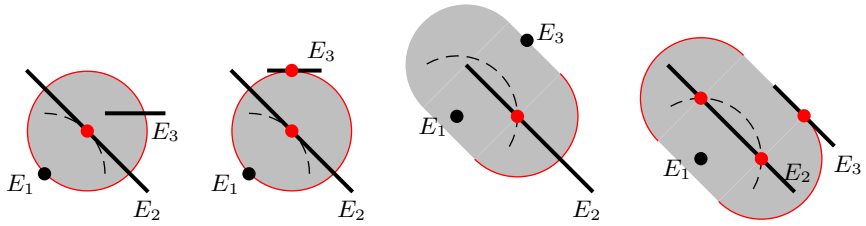

Fig. 4. Sequence (E_1, E_2, E_3) with outcomes of type α, β, γ, and δ respectively for the case that Λ is minimum such that $U_2(\Lambda) \cap E_3 \neq \emptyset$. $S_1(\Lambda)$ is depicted by dashed lines. $S_1(\Lambda) \cap E_2$, $S_2(\Lambda)$ and $S_2(\Lambda) \cap E_3$ are shown in red.

α. For the minimum L^* such that E_m intersects the interior of $U_{m-1}(L^*)$, there is no critical path.

β. For the minimum L^* such that $U_{m-1}(L^*) \cap E_m = S_{m-1}(L^*) \cap E_m = \{p\}$, there is a critical path.

γ. For the minimum L^* such that $U_{m-1}(L^*) \cap E_m$ is a single point and $S_{m-1}(L^*) \cap E_m = \emptyset$, there is no critical path.

δ. For the minimum L^* such that $U_{m-1}(L^*) \cap E_m$ is a segment and $S_{m-1}(L^*) \cap E_m$ consists only of extremities of the segment, there is no critical path.

Corollary 1. *If $(E_1, \ldots, E_m)$ supports a critical path of locally longest edges in a minimum solution tree then there is a unique choice of point locations that defines the critical path.*

Description of the algorithm. We show how to compute an optimal solution by examining possible critical paths, using the following corollary.

Corollary 2. *If $(E_1, \ldots, E_m)$ supports a critical path of locally longest edges in a minimum solution tree then this choice is part of an optimal solution.*

Note that if we had an oracle giving us a $(E_1, \ldots, E_m)$ that supports a critical path of locally longest edges in the optimal solution, then we could determine the choice of points on these elements in the optimal solution. We could then replace the segments in the sequence by fixed points and solve the rest of the

problem separately. This eventually would allow us to replace all segments by fixed points, and then to solve the problem by finding an associated MBST.

Lacking an oracle we determine these sequences *by complete enumeration* of *all possible sequences* $(E_1, \ldots, E_m)$, where E_1 and E_m are fixed points or segments, and $E_2, \ldots, E_{m-1}$ are segments. There are $O((n+k)^2 \cdot k! \cdot k)$ such sequences. 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 whether the edge-length for this path is best so far. If not, we discard this path. Otherwise we recurse on the updated set of sequences and points. That is, we prune these sequences in the enumeration as we find them. Once we have gone through the complete list of possible sequences, we will have found the critical path with shortest locally longest edges. Then we 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 greater than the ones in the critical path just found. Once the instance does not contain any more segments, we connect all remaining connected components with a greedy algorithm, in polynomial time.

We remark that it is crucial to determine critical paths with progressively decreasing edge lengths since positions of points on segments should only be determined using the locally longest incident edges.

The enumeration in our algorithm described above is superexponential in the number k of segments, which is not surprising since we have shown the problem with no fixed points is NP-hard. For constant k the problem is, however, polynomial in the number n of points, as our running time analysis will show.

The (multiple recursive) enumeration results in a search tree of size $O(((n+k)^2 \cdot k! \cdot k^2)^k)$ with a $O(k)$ running time for each node in the search tree. Thus the total time complexity is $O(((n+k)^2 \cdot k! \cdot k^2)^k \cdot k)$.

4 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.

Lemma 3. *Given a set of uncertainty regions that are unit disks $D_1, \ldots, D_n$ with centers $p_1, \ldots, p_n$, let L be the largest edge of a bottleneck spanning 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 BCU Problem. In other words, if OPT denotes the smallest radius α for any choice of $\ell_i \in D_i$, then $L/2 \leq OPT+1$. This approximation can be computed in polynomial time.*

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,

similarly, the maximum length, L, of any edge of a MBST on the $\{p_i\}$ must be at most 2 greater than the maximum length edge in the MBST on the ℓ_i. The result follows. □

Our approximation for the BCU 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). However, it is a constant factor approximation for *non-overlapping* unit disks.

5 Other Related Results and Conclusions

In this work, we also studied a closely related problem, *Worst Case Connectivity with Uncertainty* (WCU):

The WCU Problem. Find the minimum value α such that for *any* choice of points P, the connectivity graph G_α of P is connected.

We were able to show a simple approximation algorithm for WCU that is within an additive factor of 1 and a muliplicative factor of 2 when the uncertainty regions are unit disks (result omitted due to space restriction). Although we have been able to obtain several NP-hardness results for BCU, we do not have any complexity lower bounds for WCU which, a priori, seems harder. It is an interesting open question to improve our approximation algorithms for both these problems.

It would be interesting to show NP-hardness results for the BCU problem for other uncertainty regions such as disks. It is also possible that techniques from convex optimization could be used to design approximation algorithms for BCU for, say, line segments or squares. From the perspective of fixed parameter tractability, we observed that BCU is in FPT when the instance consists of n fixed points and k pairs of points; the parameter is k. However, we conjecture that BCU for the case of line segments is W[1]-hard and hence our exact algorithm is unlikely to be improved upon significantly.

In conclusion, our work on connectivity problems for uncertainty regions motivated by wireless network scenarios suggests that this area provides a rich collection of problems for further investigation.

References

1. H. Alt, E. Arkin, H. Brönnimann, 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. 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.
6. 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.
7. B. Fuchs. On the hardness of range assignment problems. In *Proc. 6th Italian Conf. Alg. and Compl. (CIAC)*, pages 127–138, 2006.
8. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, 1979.
9. J. Gudmundsson and C. Levcopoulos. A fast approximation algorithm for TSP with neighborhoods. *Nord. J. Comput.*, 6(4):469–, 1999.
10. 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.
11. 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.
12. D. Lichtenstein. Planar formulae and their uses. *SIAM J. Comput.*, 11(2):329–343, 1982.
13. 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.
14. G. Parker and R. L. Rardin. Guaranteed performance heuristics for the bottleneck traveling salesman problem. *Operations Research Letters*, 2(6):269–272, 1984.
15. P. Rosenstiehl and R. E. Tarjan. Rectilinear planar layouts and bipolar orientations of planar graphs. *Disc. Comp. Geom.*, 1:343–353, 1986.
16. 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.

A Appendix

A.1 NP-hardness of BCU with line segments and point pairs

Section 2 provided an overview of the reduction from Planar 3-SAT to the BCU problem for uncertainty regions of vertically aligned pairs of points unit distance apart with integer coordinates. The same reduction can be seen to apply to unit length vertical segments in place of the point pairs. Here we provide a more detailed description of the gadgets. We start with a formal definition of Planar 3SAT. Let $\Phi = (X, C)$ be an instance of 3-SAT, with variables $X = \{x_1, \ldots, x_n\}$ 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 is known to be NP-complete and has several formulations [8, 12].

The clause gadget. Figure 5 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 gadgets for 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 6 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 horizontal variable segments lying below. The width of the clause gadget can be increased by adding more uncertainty pairs as indicared by the figure.

Consider the three uncertainty pairs with brown and green points in Figure 6. 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. Note that the clause gadget never connects two variable gadgets. 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.

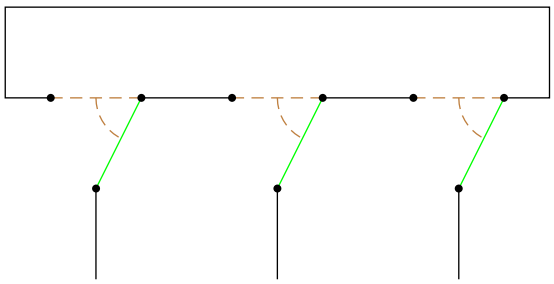

Fig. 5. Schema of the clause gadget. When at least one of the green switches in used, its gate is open and the entire clause gadget can be connected to the rest of the graph.

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 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.

Note that the subgraph of G_1 arising from a variable gadget must be internally connected, as it is not possible to join parts of it using paths running through subgraphs arising from clause gadgets or other variable gadgets. The variable gadget consists of constructs of type I, II and III. In order for G_1 to be connected, 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 gadget embedded above the variable gadget (i.e., in the grid embedding of $H(\Phi)$, the horizontal segment representing the clause lies above the horizontal segment representing the variable), then the connection from the corresponding clause gadget to this variable gadget (to be described in the next

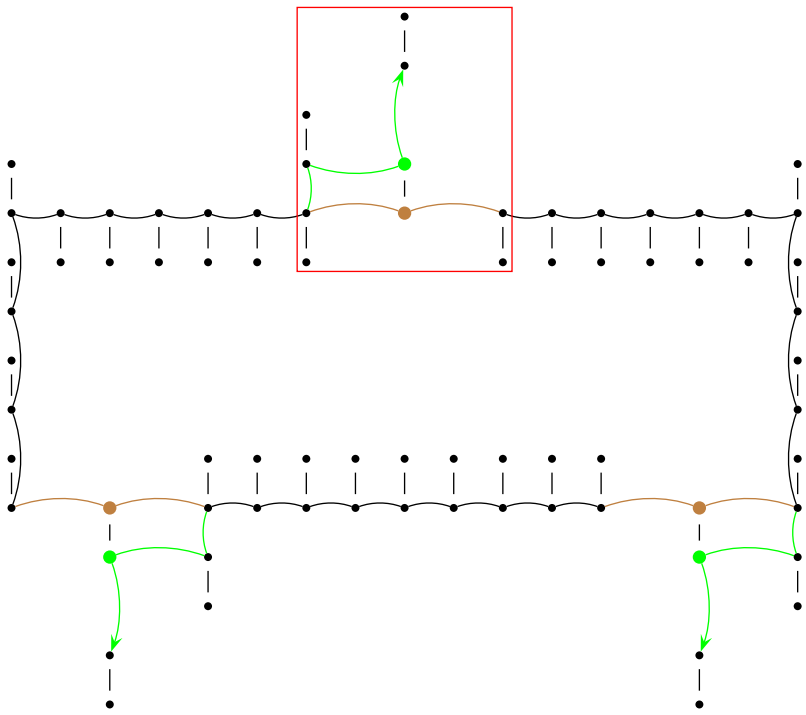

Fig. 6. An example clause gadget. As shown, aside from vertical sequences of uncertainty pairs leading to variable gadgets, there are no other uncertainty pairs in the vicinity of the clause gadget. The graph G_1 can be connected only if at least one of the green points is chosen. The choice of a green point is intended to mean that the attached literal contributes a "true" to the clause.

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. Considering that pairs in the variable gadget cannot be connected via a path though another variable or a clause gadget, we conclude that the variable gadget is designed in such a way such that the pairs in the gadget are connected iff there is a consistent truth assignment to this variable in all the clauses.

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 depends on the number of clauses containing this variable.

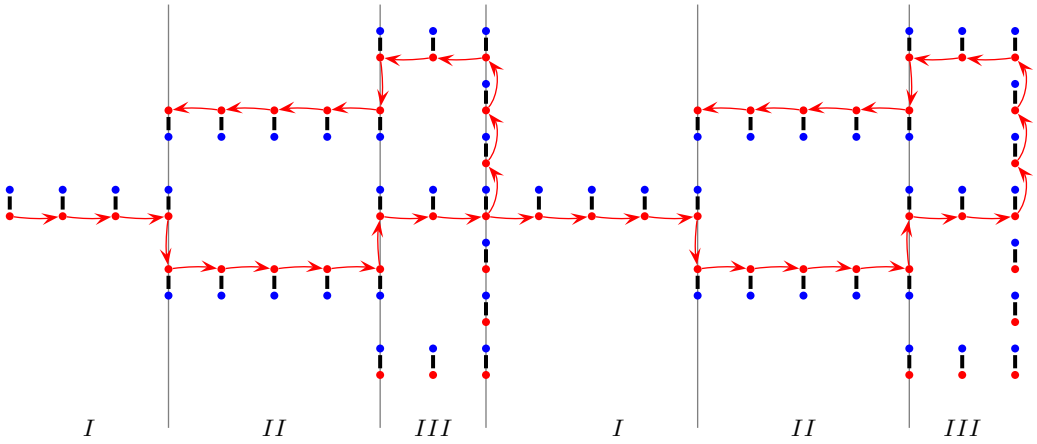

Fig. 7. An example variable gadget.

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 represented by 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 (left) connect pairs of variable gadgets and (right) clause and variable gadgets. We observe the following properties of the two connectors: In a clause-variable connector, 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.

Consider again the variable gadget. Note that if the top of construct I has an odd integer coordinate, the top of construct II has an even integer coordinate. In order to allow connections to clause gadgets from both of such constructs, we need some additional flexibility in the switches in the clause gadget. Consider again Figure 6. If necessary, switch constructs can be shifted by one unit towards the exterior of the gadget to allow such connections. For example, the switch on the top of the gadget in the figure can be shifted by moving all the uncertainty pairs in the red box up by one unit. This change preserves the properties of the clause gadget outlined earlier.

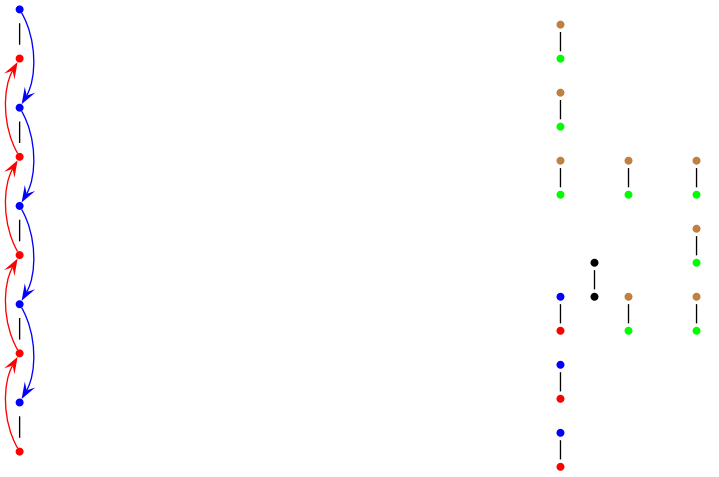

Fig. 8. (Left): A connector between clause and variable gadgets. (Right): A connector between variable gadgets.

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 horizontal 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. Let P be the set of points chosen in all the uncertainty pairs and let us consider the graph G_1 of P. In G_1, the variable gadgets are all internally connected and connected to each other using 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.

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. Since two different variable gadgets can never be connected to each other through a clause gadget, all the variable gadgets have to be internally connected and also connected to each other. This structure of G_1 gives a truth assignment for each variable depending on whether the blue or red points are chosen inside the gadget. Since every clause gadget is connected to at least one variable gadget, the truth assignment satisfies every clause. It is easy to see that this truth assignment satisfies the Planar 3-SAT instance Φ.

By reduction from Planar 3-SAT, BCU for non-overlapping uncertainty regions that are pairs of vertically aligned points one unit apart with integer coordinates is NP-hard. Essentially the same proof holds when the pairs are replaced by vertical line segments of unit length.

Best approximation We observe that for uncertainty regions which are pairs of points, there is no approximation algorithm, polynomial in the size of the input, with approximation ratio less than $\sqrt{5}/2$, unless $P = NP$. Indeed, we have provided problem instances where the uncertainty regions are vertically aligned pairs of points separated by unit distance such that a Bottleneck Spanning Tree of maximum edge length 2 ($\alpha = 1$) can be found if and only if $P = NP$. But two points on the integer grid, if further apart than distance 2 must be at least distance $\sqrt{5}$ from one another. Hence if we had a polynomial time approximation to the solution with a ratio less than $\sqrt{5}/2$ then we could use the approximation to find a Bottleneck Spanning Tree with maximum edge length not greater than 2, a contradiction (unless $P = NP$).

Uncertainty on line segments. As previously noted, we can also prove the BCU problem is NP-hard for vertical unit segment on an integer grid. However, the hardness of approximation result does not hold, because we can use edges of length arbitrarily close to 1.

A.2 NP-hardness of the BCU problem with unit square uncertainty regions

We again reduce the problem Planar-3-SAT to the BCU problem with square uncertainties.

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

Planar Graph Embedding on a Square Grid. 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.

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.

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

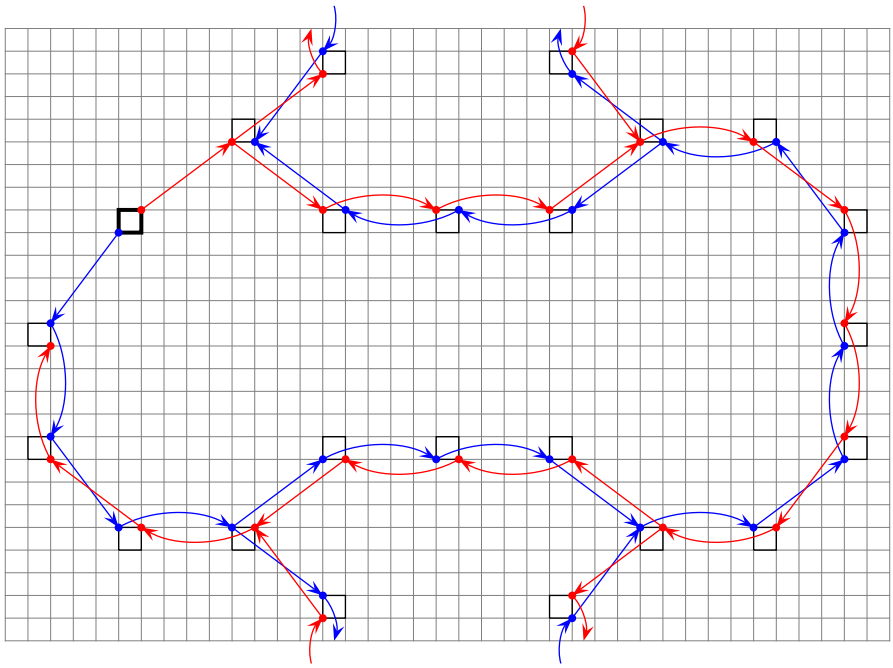

Fig. 9. An example variable gadget.

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

The variable gadget, as shown in the figure, can be 5-connected in two ways: by choosing the red points in each square, or by choosing the blue points in each square. We call the bold square in the figure the *reference* square. The reference square can only be connected to at most one square among the other squares in the variable gadget. If the blue point is chosen to make this connection in the reference square, then we say that the variable associated with this gadget is True; if the red point is chosen, we say that the variable is False.

Connections to clause gadgets drawn in place of horizontal line segments representing clauses that contain this variable or its negation are made via the four extreme top and bottom squares. Of these, the top left and bottom right connect to clauses containing this variable and the top right and bottom left connect to clauses containing the negation of this variable. The variable gadget can be easily modified to allow additional connections to clause gadgets above or below this variable gadget by being increased in width.

The clause gadget. For each clause, we create the gadget shown in Figure 10. We call the bold square in the clause gadget the *core* square. For each of the 3 literals (a variable or its negation), there is a sequence of squares in the gadget, called an *arm*. Observe that the core square can be 5-connected to only a single arm, and once this arm is selected, the choice of points in its squares is *fixed* in

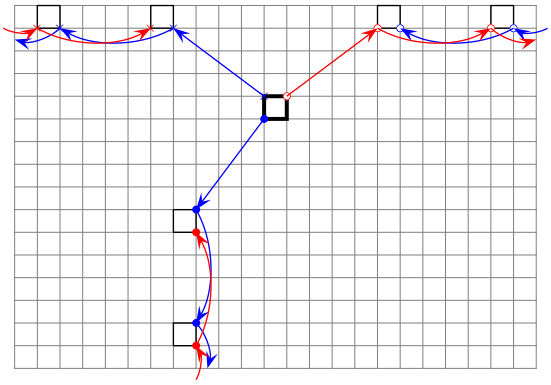

Fig. 10. The clause gadget.

order for the squares to be 5-connected. The choice of points within the other arms' squares is *free* since they do not need to connect to the core point. These squares can be made connected to each other.

Linking the gadgets. Here we explain how variable gadgets can be linked to clause gadgets and how variable gadgets can be linked to each other.

In the gadget shown in Figure 10, in order to connect to more than one variable gadget positioned beneath the clause gadget, it is necessary to change the location of the red and blue points in the square regions from being horizontally aligned to vertical. This can be accomplished using the corner gadget, shown in Figure 11. Other useful versions of this gadget can be made by taking its horizontal and vertical mirror images. Likewise, the corner gadget allows to connect to variable gadgets above the clause gadget. To connect the lower arm of the clause gadget to a variable gadget above the clause gadget, two corner gadgets are necessary.

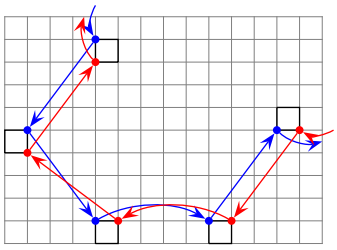

Fig. 11. A corner gadget.

To complete the connection between vertex and clause gadgets, it remains to explain how to connect pairs of square uncertainty regions, one in the vertex gadget and the other in an extension of one the arms of the clause gadgets. We would like to propagate truth assignments (choice of red or blue point in the variable gadget) consistently along such connectors. When the distance between the points to be joined in the two squares is a multiple of 5 and these points are vertically aligned, the connection can be made in the way analogous to that shown in Figure 8(Left). Otherwise, when the two squares to be connected are placed arbitrarily in space, the construction of such connectors can be easily accomplished but is quite tedious to describe. We leave this construction to the reader.

To connect variable gadgets together, we add $n-1$ *loose* connections between the n variable gadgets by using a sequence of squares that allow the variable gadgets to be 5-connected to each other, regardless of the choice of point in the squares of the gadget. Figure 12 shows an example of such a connection.

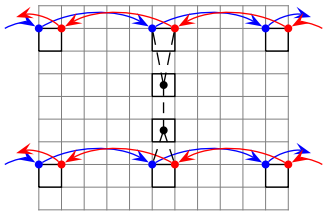

Fig. 12. An example of a loose connection.

Correctness of the Reduction. We now argue that there exists a choice of point in each of the square uncertainty regions such that the connectivity graph for $\alpha = 5/2$ is connected if and only if the corresponding Planar 3-SAT instance is satisfiable.

Suppose that there is a satisfying assignment to the Planar 3-SAT instance. Then, in each variable gadget we choose the blue corner of the reference square when that variable is True and the red corner when that variable is False in the satisfying assignment. By choosing points in the other squares of the variable gadgets to be the same colour as that of the reference square, we make the variable gadget connected. The choice of red or blue point is then propagated to a clause gadget satisfied by this assignment via the connectors. Note that the truth value is correctly inverted by the variable gadget when connecting a negated variable to a clause that includes this literal. Since all the clauses are satisfied, the core square in each clause gadget is connected to the variable that satisfied the clause. The arms that don't connect to the core square are connected to their respective variable gadgets. At this point we have created n

trees, one for each variable. The $n-1$ connectors between variable gadgets make $G_{5/2}$ connected.

Let P be any choice of points in the square regions for which the corresponding graph $G_{5/2}$ is connected. Since $G_{5/2}$ is connected, each clause gadget is connected to exactly one variable gadget. Because variable gadgets can never be connected to each other via a clause gadget, each variable gadget must be internally connected. The choice of red or blue point in the reference square of each variable gadget assigns the satisfying truth assignment to the variable associated with that variable gadget in the 3-SAT instance.

By reduction from Planar 3-SAT, BCU for non-overlapping unit square uncertainty regions whose corners can be given integer coordinates is NP-hard.

A.3 Our Results for the WCU Problem

Lemma 4. *Given a set of uncertainty regions that are unit disks $D_1, \ldots, D_n$ with centers $p_1, \ldots, p_n$, let L be the largest edge of an MBST on $\{p_i\}$. Then choosing $r = L/2 + 1$ always results in the connectivity graph being connected and is at worst an OPT+1 approximation to the WCU Problem.*

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 an 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 $\ell_i = p_i \ \forall i$ hence the minimum α is $L/2$. Hence OPT $\geq L/2$ and the lemma is established. □

Lemma 5. *Given a set of uncertainty regions that 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 WCU problem.*

Proof. Note that OPT+1 $\leq$ 2OPT as long as OPT ≥ 1 so, by Lemma 4, 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 connectivity graph connected. It follows that OPT ≥ 1 and the lemma is established. □

Unfortunately our approximation for the BCU 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 WCU Problem, Lemma 5 shows that the simple broadcast-from-center heuristic can be no worse than a 2-approximation. However, we thus far only found examples showing that the approximation can be (asymptotically) as bad as a $\sqrt{2}$-approximation, as the next Lemma asserts.

Lemma 6. *Given any $L > 2$, there is an instance of the WCU 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 4 can be as bad as a factor $\sqrt{2} - \epsilon$ approximation for arbitrarily small ϵ.*

Proof. We distribute an even number of unit disks with centers equally spaced along a very large (relative to the unit disks) 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 contains a large number of highly overlapping disks in addition to the disks whose centers lie along C. See Figure 13 for a sketch. The

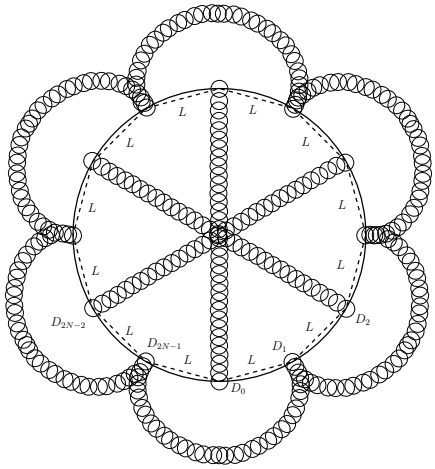

Fig. 13. The construction begins with an even number of equally spaced unit disks with centers along a very large circle C.

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, 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 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 centers 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 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 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 13 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 14. Regardless of the choice of the $\ell_{i_j} \in D_{i_j}$ it is clear

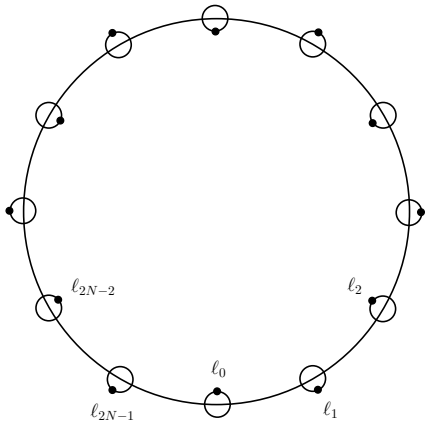

Fig. 14. The distance between ℓ_i and ℓ_{i+1} (in the cyclical ordering) is then just slightly less than $\sqrt{L^2+4}$.

that $\bigcup\{D(\ell_i;\alpha)\} \cup \bigcup\{D(\ell_{i_j};\alpha)\}$ is connected if 2α is the 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 4 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. □

While the broadcast-from-center heuristic is not a constant factor approximation to the BCU 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 BCU 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 find an 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, 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 15.

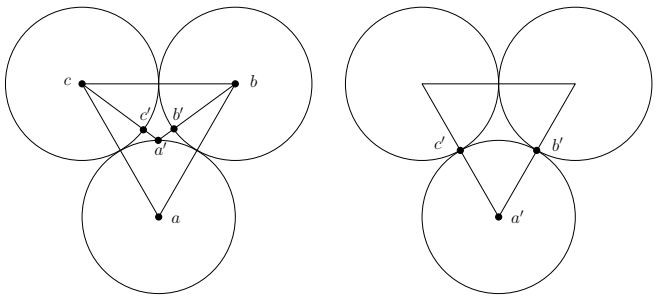

Fig. 15. The $B\alpha$ Problem for three (almost) tangent unit disks. OPT, as shown on the left is given by the choice of broadcasting locations (a', b', c'), not quite on an equilateral triangle, with MBST cost $(\sqrt{1 + (\sqrt{3} - 1)^2} - 1)/2 \approx .12$, while the "cinch-up" heuristic, on the right, chooses broadcasting locations (a', b', c') with cost 0.5.