Minimizing the Stabbing Number of Matchings, Trees, and Triangulations*

Sándor P. Fekete† Marco E. Lübbecke‡ Henk Meijer§

Abstract

The (axis-parallel) stabbing number of a given set of line segments is the maximum number of segments that can be intersected by any one (axis-parallel) line. This paper deals with finding perfect matchings, spanning trees, or triangulations of minimum stabbing number for a given set of points. The complexity of these problems has been a long-standing open question; in fact, it is one of the original 30 outstanding open problems in computational geometry on the list by Demaine, Mitchell, and O'Rourke.

The answer we provide is negative for a number of minimum stabbing problems by showing them $\mathcal{NP}$-hard by means of a general proof technique. It implies non-trivial lower bounds on the approximability. On the positive side we propose a cut-based integer programming formulation for minimizing the stabbing number of matchings and spanning trees. We obtain lower bounds (in polynomial time) from the corresponding linear programming relaxations, and show that an optimal fractional solution always contains an edge of at least constant weight. This result constitutes a crucial step towards a constant-factor approximation via an iterated rounding scheme. In computational experiments we demonstrate that our approach allows for actually solving problems with up to several hundred points optimally or near-optimally.

ACM Classification: F.2.2 Nonnumerical Algorithms and Problems.

AMS Classification: 68Q17, 68U05, 90C27.

Keywords: Stabbing number, crossing number, matching, spanning tree, triangulation, complexity, linear programming relaxation, iterated rounding.

1 Introduction

Objective Functions. Typical problems in combinatorial optimization, algorithmic graph theory, or computational geometry deal with minimizing the length of a desired structure: Given a set of points, find a set of line segments of small total length, such that a certain structural condition is maintained. Among the most popular such structures are spanning trees, perfect matchings, or (in a

*An extended abstract appeared in the *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms [10].*

†Institute of Mathematical Optimization, Braunschweig University of Technology, Pockelsstraße 14, D-38106 Braunschweig, Germany. Email: s.fekete@tu-bs.de.

‡Institut für Mathematik, Sekr. MA 6-1, Technische Universität Berlin, Straße des 17. Juni 136, D-10623 Berlin, Germany. Email: m.luebbecke@math.tu-berlin.de. Visits to Kingston and Stony Brook were supported by a DFG travel grant.

§Department of Computing and Information Science, Queen's University, Kingston, Ontario, K7L 3N6, Canada. Email: henk@cs.queensu.ca. Partially supported by NSERC while visiting Braunschweig.

planar geometric setting) triangulations. However, some geometric scenarios motivate other objective functions; one such alternative for measuring the quality of a structure is the total turn cost between adjacent line segments; e.g., see [3].

When dealing with structural or algorithmic properties, one can be more interested in yet another objective function called the *stabbing number*: In order to unify definitions for different structures and to allow for a consistent notation throughout this paper, we describe this as a property of a set of line segments: For a given set of line segments, this is the maximum number of segments that are encountered (in their interior or at an endpoint) by any infinite line; if we consider only axis-parallel lines, we get the *axis-parallel stabbing number*. When focusing on the number of objects defined by the line segments, we may consider the closely related *crossing number* that arises from the number of connected components of the set of line segments that we have to cross along a line or a ray to reach infinity. In the absence of connected components of collinear segments (which is the case for matchings), the crossing number coincides with the stabbing number. When considering structures like triangulations, the crossing number is precisely one more than the maximum number of triangles intersected by any one line.

Related Work. Stabbing problems have been considered for a number of years. The complexity of many algorithms in computational geometry is directly dependent on the complexity of ray-shooting, which depends directly on the stabbing number. Agarwal [1] describes several applications of spanning trees with low stabbing number, among them ray-shooting and implicit point locations queries (which by themselves have applications in polygon containment, implicit hidden surface removal, polygon placement, etc.). One of the theoretically best performing data structures for ray tracing in two dimensions is based on a triangulation of the scene; see Hershberger and Suri [13]. Agarwal, Aronov, and Suri [2]. investigate the stabbing number of triangulations in three dimensions, where the stabbed objects are simplices. See also Aronov and Fortune [5] for this problem, and Aronov et al. [4] for a recent experimental study. Held, Klosowski, and Mitchell [12] investigate collision detection in a virtual reality environment; again, we have a dependency on the stabbing number.

Extremal properties of crossing numbers were considered by Welzl [26] and by Matoušek [17], who showed that any planar set of n points has a spanning tree with a crossing number of $O(\sqrt{n})$, and there are examples requiring a crossing number of $\Omega(\sqrt{n})$. Another variant is studied by de Berg and van Kreveld [6]: The stabbing number of a decomposition of a rectilinear polygon P into rectangles is the maximum number of rectangles intersected by any axis-parallel segment that lies completely inside of P; they prove that any simple rectilinear polygon with n vertices admits a decomposition with stabbing number $O(\log n)$, and they give an example of a simple rectilinear polygon for which any decomposition has stabbing number $\Omega(\log n)$. They generalize their results to rectilinear polygons with rectilinear holes. Shewchuk [23] shows that in d dimensions, a line can stab the interiors of $\Theta(n^{\lceil d/2 \rceil})$ Delaunay d-simplices. This implies, in particular, that a Delaunay triangulation in the plane may have linear stabbing number. More recently, Tóth [25] showed that for any subdivision of size n in a d-dimensional Euclidean space, $d \geq 2$, there is an axis-parallel line that stabs at least $\Omega(\log^{1/(d-1)} n)$ boxes, which is best possible.

Despite of this interest, nothing is known about the computational complexity of stabbing problems. In fact, settling the complexity of Minimum Stabbing Number for spanning trees has been one of the original 30 outstanding open problems of computational geometry on the list by Mitchell and O'Rourke [19] (an up-to-date list is maintained online by Demaine, Mitchell, and O'Rourke [7]). In addition, except the factor $O(\sqrt{n})$ coming from Welzl's work, there are no results available about

approximation algorithms or lower bounds on the stabbing number.

Our Contributions. While previous work on stabbing problems has focused on extremal properties, our paper has a strong algorithmic flavor. We describe a general proof technique that shows $\mathcal{NP}$-hardness of minimizing the stabbing number of perfect matchings, triangulations, and spanning trees. For the case of matchings we show that it is also hard to approximate the minimum stabbing number.

On the other hand we present a mathematical programming framework for solving stabbing problems. Characterizing solutions to stabbing problems as integer programs with an exponential number of cut constraints, we describe how the corresponding linear programming (LP) relaxations can be solved in polynomial time, providing empirically excellent lower bounds. Exploiting geometry, we show that an optimal fractional matching (or spanning tree) always contains an edge of constant weight, allowing an iterated rounding scheme similar to the one developed by Jain for the generalized Steiner network problem [14]: Compute a heuristic solution by solving a polynomial sequence of LPs. We have reason to believe that this heuristic solution is within a constant factor of the optimum. Finally, we show that our mathematical programming approach is also practically useful by demonstrating that we can optimally solve stabbing problems for well-known benchmark instances of point sets up to several hundred points.

Our results in detail:

- We prove that deciding whether a point set has a perfect matching of axis-parallel stabbing number 5 is an NP-complete problem; we also extend this result to general stabbing number.
- We prove that finding a triangulation of minimum axis-parallel crossing number is NP-hard.
- We give an NP-completeness proof for finding a spanning tree of axis-parallel or general stabbing number, and sketch hardness proofs for axis-parallel or general crossing number.
- We describe an LP-based class of lower bounds that can be evaluated in polynomial time.
- We give results on the structure of fractional vertices of the resulting LP-relaxation: For matching, we show that there always is an edge with weight at least $1/5$, while for spanning trees, there always is an edge with weight greater than $1/3$. This allows an iterated rounding technique, similar to the one developed by Jain for generalized Steiner network problems; we believe that the resulting polynomial algorithm produces a constant-factor approximations.
- We describe the results of a computational study. Using a diverse set of benchmark instances (based on TSPLIB, Solomon's vehicle routing problems, and two different types of random instances) we are able to compute optimal and near-optimal solutions for instances up to several hundred points. This demonstrates that our LP-based approach is good not only in theory (where we get a polynomial running time based on the ellipsoid method), but also for actually solving instances in practice (where we use the simplex method). Results indicate far better approximation quality than the theoretically possible factors of 5 or 3, respectively.

It should be noted that our positive (LP-based) results do *not* make any assumptions on the structure of the point set: They can be used for point sets in degenerate as well as in general position, and can be applied to any family of stabbing lines that can be evaluated by considering a subset of polynomially many representatives. On the other hand, the point sets constructed in our hardness proofs make use of collinear points.

The rest of this paper is organized as follows. After some basic definitions and notation in Section 2, we give details of our various hardness proofs in Section 3. In Section 4, we describe our LP-based approach for constructing bounds. Section 5 presents an iterated rounding technique for matching and spanning tree problems; we believe that the resulting algorithms are constant-factor approximations. Section 6 presents a detailed computational study on perfect matchings of low stabbing number. Final concluding thoughts and miscellaneous results and problems are presented in Section 7.

2 Preliminaries

Given a set of line segments L in the plane, the *stabbing number* of a line ℓ is the number of segments of L that are intersected by ℓ. The *stabbing number of* L is the maximum stabbing number over all lines ℓ; the *axis-parallel stabbing number of* L is the maximum stabbing number over all axis-parallel lines ℓ. In this paper, the set L will arise as a perfect matching, spanning tree, or triangulation of a given set P of n points in the plane, and our objective is to find such a structure of minimum stabbing number. Any reference to matching always means perfect matching. Therefore, when dealing with matchings, we assume that n be even, if necessary by omitting one of the points.

We denote by *St-Mat*(P) the minimum stabbing number among all matchings of P, by *St-Tre*(P) the minimum stabbing number of all spanning trees of P, and by *St-*$\Delta(P)$ the minimum stabbing number of all triangulations of P. In this notation we indicate by an additional subscript the cases in which we are interested in the respective minimum axis-parallel stabbing numbers only, that is, we use *St-Mat*$_2(P)$, *St-Tre*$_2(P)$, and *St-*$\Delta_2(P)$ in these cases.

Instead of considering the number of line segments in L encountered by a line ℓ, we will also consider the number of connected components of $L \cap \ell$. Then we speak of the *crossing number*. For matchings, trees, and triangulations, we use the analogous abbreviations *Cr-Mat*(P), *Cr-Tre*(P), and *Cr-*$\Delta(P)$, and their subscripted counterparts *Cr-Mat*$_2(P)$, *Cr-Tre*$_2(P)$, and *Cr-*$\Delta_2(P)$ for the axis-parallel crossing numbers. Note that stabbing and crossing number coincide for planar matchings.

3 Complexity

In this section we prove $\mathcal{NP}$-hardness of computing the minimum stabbing number of matchings and computing the minimum crossing number of triangulations; for spanning trees, proofs are analogous, and we only give a sketch of the proof. Our technique is rather general and should be applicable to other structures and variants as well.

3.1 Perfect Matchings

Theorem 1 *Deciding whether St-Mat*$_2(P) \leq 5$ *is strongly* $\mathcal{NP}$*-complete.*

Proof. Clearly, the problem is in $\mathcal{NP}$. We show completeness using a reduction from 3SAT [11]. Assume we have a Boolean expression denoted by $B(x_0, x_1, \ldots, x_{n-1})$ with n variables and k clauses of three literals each. We construct a set of points P that has a perfect matching M of stabbing number 5 if and only if the Boolean expression can be satisfied.

Consider the overall layout of P as shown in Figure 1. We make critical use of the collinearity of points, using up all of the available stabbing number of 5 in a particular direction. Thus we are able

to construct "barriers" which avoid any interference between the different gadgets.

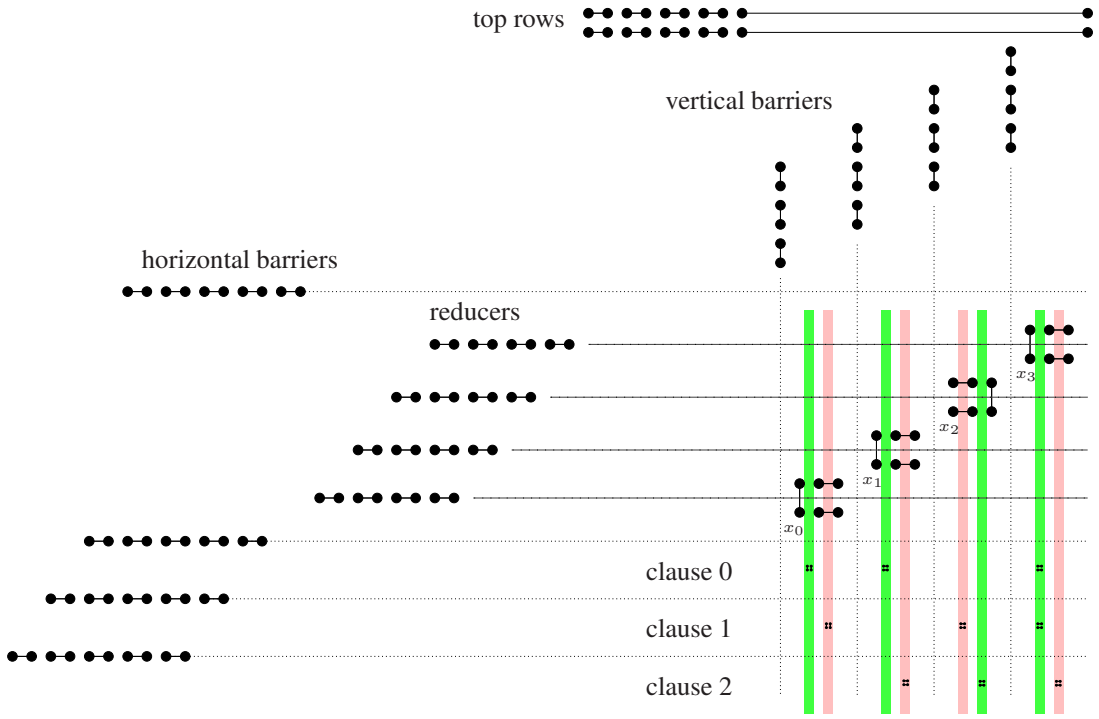

Figure 1: Overall layout of the construction for *St-Mat*$_2(P)$. Shown is the layout for the 3SAT instance $(x_0 \vee x_1 \vee x_3) \wedge (\overline{x_0} \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$, with a truth setting of x_0 =*true*, x_1 =*true*, x_2 =*false*, x_0 =*true*.

At the top of the layout are two groups of 10 points. The points in a group of 10 have the same y-coordinates. We call these two groups the *top rows*. The i-th point in the first top row has the same x-coordinate as the i-th point in the second top row. Below the top rows are n groups of 6 points. All points in a group of 6 have the same x-coordinates, as shown in the figure. The vertical lines through these groups of 6 points separate variables and separate the variables from other points left of the variables. We call each such group a *vertical barrier*. The points in barrier i are far enough below the points in barrier $i+1$ to ensure that horizontal lines through the vertical barriers have stabbing number at most 5. The barriers lie between the last and second last points of the top rows. To the left of the top rows and below the vertical barriers are $k+1$ groups of 10 points. Each point in a group of 10 has the same y-coordinates. The horizontal lines through these groups of 10 points separate clauses. We call each such group a *horizontal barrier*. The points in barrier i are far enough to the right of the points in barrier $i+1$ to ensure that vertical lines through the horizontal barriers have stabbing number at most 5. The horizontal barriers are used to separate clauses from each other, to separate the clauses from the variables, and to separate variables from other points above the variables. Between and to the right of the top two horizontal barriers are groups of 8 points. Each point in a group of 8 has the same y-coordinates. We call each such group a *reducer* of a variable. The points in reducer i are far enough to the right of the points in reducer $i+1$ to ensure that vertical lines through the reducers have stabbing number at most 5.

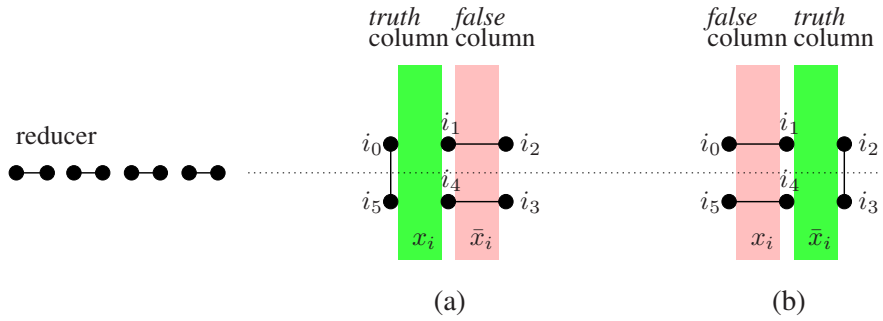

Figure 2: Variable x_i with $x_i = \textit{true}$ in (a) and $x_i = \textit{false}$ in (b).

Figure 2 shows a set of 6 points, numbered i_0 to i_5 in clockwise order, that represents a variable. The variables are separated by vertical barriers, and are placed below the vertical barriers, between top two horizontal barriers and to the right of the reducers. Variables are placed such that the y-coordinate of the horizontal line through the reducer of a variable is in-between the y-coordinates of points i_0 and i_5 of that variable. The collection of vertical lines between points i_0 and i_1 of variable x_i is called the x_i-*column* of the variable. The collection of vertical lines between points i_1 and i_2 of variable x_i is called the $\bar{x}_i$-*column* of the variable.

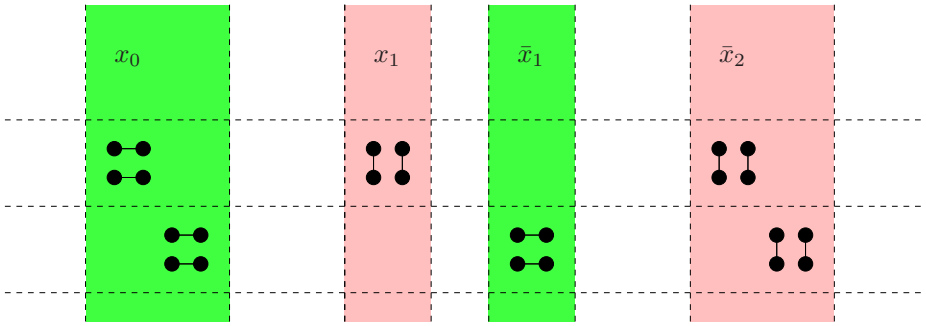

Figure 3: Clauses $(x_0 \vee x_1 \vee \bar{x}_2)$ and $(x_0 \vee \bar{x}_1 \vee \bar{x}_2)$ with $x_0 = x_2 = \textit{true}$ and $x_1 = \textit{false}$.

The horizontal and vertical barriers create kn locations for literals. The literals are groups of 4 points representing the presence of a variable in a clause. Each group of 4 points forms an axis-parallel square. If a literal x_i appears in the clause c_j, we place it in the x_i column of clause c_j. If a literal $\bar{x}_i$ appears in the clause c_j, we place it in the $\bar{x}_i$ column of clause c_j. The three literals of a clause are put on the same horizontal lines. The literal in column x_i of clause c_j is to the left of the literal in column x_i of clause c_h for $j < h$. Similarly, the literal in column $\bar{x}_i$ of clause c_j is to the left of the literal in column $\bar{x}_i$ of clause c_h for $j < h$. Figure 3 shows the literals of two clauses.

We first assume that $B(x_0, x_1, \ldots, x_{n-1})$ is satisfiable, and show that P has a matching M of stabbing number 5. We connect point i to point $i + 1$ in each of the two top rows for $i = 0, 2, 4, 6, 8$. We connect point i to point $i + 1$ in each vertical barrier for $i = 0, 2, 4$. We connect point i to point $i + 1$ in each horizontal barrier for $i = 0, 2, 4, 6, 8$. We connect point i to point $i + 1$ in each reducer for $i = 0, 2, 4, 6$.

If the variable x_i has the value *true*, we connect the pairs (i_0, i_5), (i_1, i_2), and (i_3, i_4) of the variable. If the variable x_i has the value *false*, we connect the pairs (i_0, i_1), (i_2, i_3), and (i_4, i_5). Notice that if x_i is *true*, then any vertical line in the x_i-column stabs 2 edges in the top rows of M, and a vertical line in the $\bar{x}_i$-column stabs 2 edges in the top row and 2 edges in the rectangle of the variable. If x_i is *false*, this situation is reversed. The column with vertical stabbing number 2 is called the *true* column of the variable, the column with vertical stabbing number 4 is called the *false*-column. In each literal representing the value *true* we connect the four points with two horizontal edges. In each literal representing the value *false* we connect the four points with two vertical edges.

We can now verify that M has stabbing number 5. Any vertical line in the *true*-column of variable x_i stabs two edges in the top rows and at most two edges in a literal. Any vertical line in the *false* column stabs two edges in the top rows, two edges of the variable and at most one in a literal. Any horizontal line in a clause stabs at most three literals, one of which is set to *true*. So these lines stab at most 5 edges of M. It can easily be verified that all other horizontal and vertical lines stab at most 5 edges from M.

Conversely, we assume that P has a matching of stabbing number 5. We show that B is satisfiable. The matching used in this proof is illustrated in Figures 1, 2 and 3. Because the top rows contain 10 points, these points have to be connected to each other, otherwise the stabbing number of P exceeds 5. There are several ways to connect the sets of 10 points. If we connect point i to point $i + 1$ for $i = 0, 2, 4, 6, 8$ in each row, the number of edges stabbed by any horizontal or vertical line is minimized. Therefore we may assume without loss of generality that these edges are in the matching M. Similarly if in the remainder of this proof there are several ways to connect a set of points, and one of these ways has the minimal stabbing number for all stabbing lines, we will use these connections.

Thus, we can connect point i to point $i + 1$ for $i = 0, 2, 4, 6, 8$ in each horizontal barrier. For the same reason, we can connect point i to point $i + 1$ for $i = 0, 2, 4, 6$ in each reducer. Because vertical lines through the vertical barriers stab two edges in the top rows, we can connect point i to point $i+1$ for $i = 0, 2, 4$ in each vertical barrier.

Each reducer contributes 4 to a horizontal stabbing number. Thus, we cannot connect the six points of a variable x_i by three vertical edges. Figure 2 shows the two remaining, essentially distinct matchings.

Of the three literals in each clause, one has to be set to true, otherwise there will be a horizontal stabber intersecting 6 edges. The true literal, say, x, must lie in a *true*-column of a variable, because vertical lines in this column have stabbing number 2. Any other literal in this column can also be set to *true*. The literals $\bar{x}$ lie in the *false*-column of the same variable, and have to be set to false. So if a matching of number 5 exists, there is a truth assignment of the Boolean expression. □

Corollary 2 *There is no α-approximation algorithm for St-Mat$_2(P)$ with $\alpha < 6/5$; in particular there is no polynomial time approximation scheme (PTAS).*

Corollary 3 *Computing St-Mat(P) is a weakly $\mathcal{NP}$-hard problem.*

Proof. We apply a perturbation technique, similar to the one in [9]. Use the same construction as for the hardness proof for the axis-parallel case. Consider the grid formed by the coordinates of the resulting point set. This grid is modified such that the interpoint distances between the points of the same gadget are $\Theta(\varepsilon^{n^2})$. Furthermore, the rest of the grid is perturbed by powers of ε, such that only axis-parallel lines can stab more than two gadgets. Now it is easy to see that only axis-parallel lines are critical. □

3.2 Triangulations

Our basic proof technique is the same as for matching. We first describe the construction of barrier gadgets, using the following terminology. A *horizontal line* is given by a set of points that are horizontally collinear. A *vertical line* is given by a set of vertically collinear points. A *row* consists of two horizontal lines, and the (empty) space between them. A *column* consists of two vertical lines, and the (empty) space between them.

Lemma 4 *Consider a row consisting of two horizontal lines l_a and l_b in P, having a and b points, respectively. If the combined number of edges on l_a and l_b is $a+b-i-2$, then a horizontal stabber between l_a and l_b encounters at least $a+b+i-2$ triangles in any triangulation of P and its crossing number is at least $a+b+i-1$.*

Proof. Suppose there are $a+b-i-2$ edges on the lines l_a and l_b. Consider the triangles stabbed by a line between l_a and l_b. The union of these triangles form a (possibly non-simple) polygon. All points on lines l_a and l_b lie on the boundary of this polygon. If we traverse the boundary of the polygon, we encounter at least $a+b+i$ vertices. Therefore l stabs at least $a+b+i-2$ triangles and $a+b+i-1$ edges. □

The lemma holds analogously for two vertical lines that form a column. When a row consists of two horizontal lines that have *Cr*-$\Delta_2(P)+1$ points altogether, we call it *full* or fully triangulated. It follows from the lemma that all *Cr*-$\Delta_2(P)-1$ edges on the lines l_a and l_b have to be present.

Theorem 5 *Finding Cr-$\Delta_2(P)$ is $\mathcal{NP}$-hard.*

Proof. Again we use a reduction from 3SAT, and the proof proceeds along the lines of the proof of Theorem 1. See Figure 4 for a schematic layout of a representing point set P for the 3SAT instance $B(x_0, x_1, x_2) = (x_0 \vee x_1 \vee \bar{x}_2) \wedge (x_0 \vee \bar{x}_1 \vee x_2) \wedge (\bar{x}_0 \vee \bar{x}_1 \vee \bar{x}_2)$. Figure 5 shows the structure of variable gadgets.

For a given Boolean expression $B(x_0, x_1, \ldots, x_{n-1})$ with n variables and k clauses of three literals each we construct a set P of points. We show that there is a value K such that B is satisfiable if and only if *Cr*-$\Delta_2(P) = 2K-1$.

In Figure 4 we have $K = 39$ and a grid of points with some well-defined holes. The maximum number of points in a horizontal or vertical line is K, and many lines have exactly K points. By Lemma 4 a full row or column in this setting has exactly $2K-2$ triangles. Gadgets are separated by full rows and columns.

Figure 5 shows two horizontally aligned rectangles of eight points each that together represent a variable x_i. We call the collection of vertical lines that stab the left rectangle the *x_i-column*, and the collection of vertical lines that stab the right rectangle the *$\bar{x}_i$-column* of the variable. The gadget works essentially the same way as that in Figure 2. Each rectangle has full rows and columns as neighbors. We indicate how this can be achieved in horizontal direction in Figure 5. By Lemma 4 we conclude that all edges of the convex hull of each rectangle are present in any triangulation of minimal crossing number. The horizontal lines that contain the top and bottom lines of the two rectangles, respectively, contain K points each; the horizontal line that passes through the middle of the rectangles contains $K-1$ points. Therefore Lemma 4 shows that exactly one edge along this horizontal line in the middle of the rectangle may be missing. We call the space that is spanned by all vertical lines that stab the rectangle with the missing horizontal edge the *true*-column of the variable x_i. The space that is

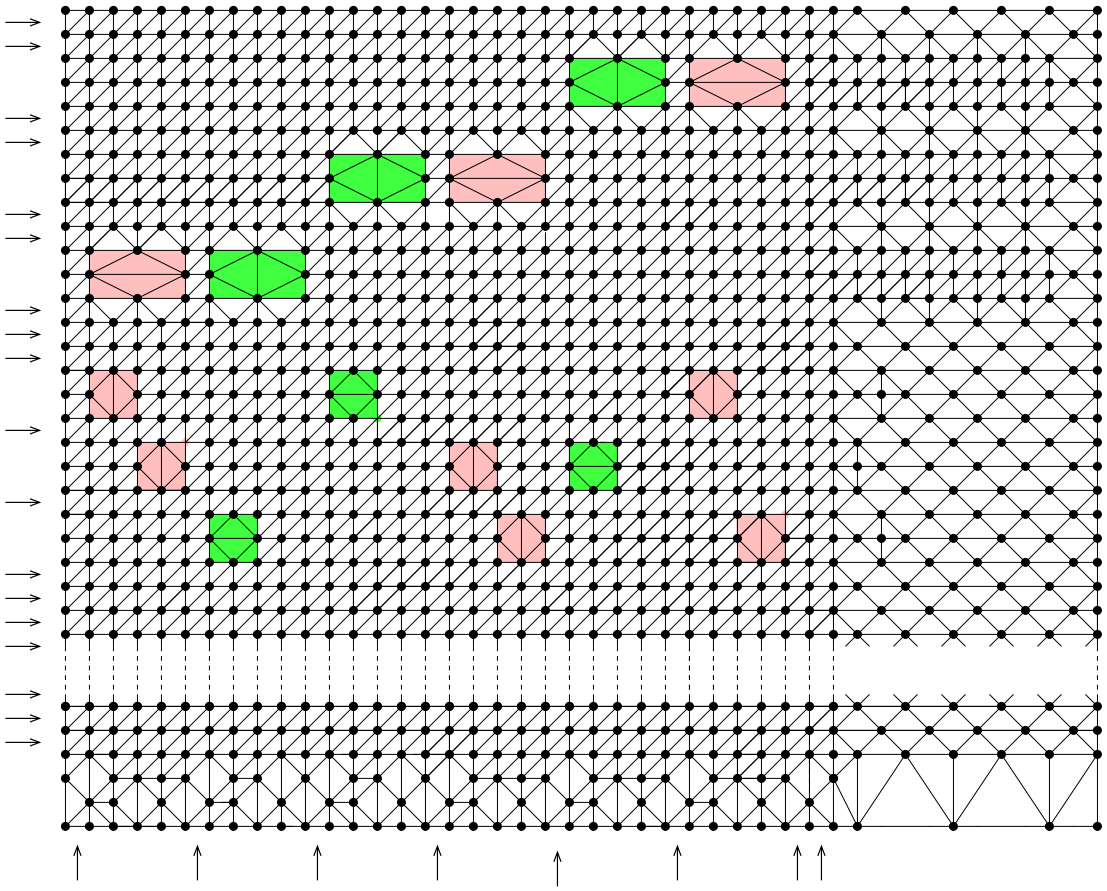

Figure 4: Overall layout for *Cr*-$\Delta_2(P)$. Clauses are $(x_0 \vee x_1 \vee \bar{x}_2)$, $(x_0 \vee \bar{x}_1 \vee x_2)$, and $(\bar{x}_0 \vee \bar{x}_1 \vee \bar{x}_2)$, with $x_0 =$ *false* and $x_1 = x_2 =$ *true*. Arrows indicate full rows and columns, light or dark shading indicates true or false variables and literals.

spanned by all vertical lines that stab the rectangle for which the middle horizontal edge is present is the *false*-column of the variable x_i. Note that the *true*-column adds exactly one less to the vertical crossing number than the *false*-column. In the overall layout, variable x_i is placed below and to the left of variable x_j for $i < j$ in such a way that variables are vertically separated from one another by a full row, and horizontally separated from one another by a full column.

Each literal is represented by a square with eight points on its boundary. We make the rectangles of the variables wide enough to accommodate the necessary number of literals. That is, each rectangle of a variable is of a width twice the number of the maximum occurrence of a literal in B. Figure 5 gives a hint at how this widening of a rectangle is done. The three literals of a clause c_j are horizontally aligned, and the two rows that are spanned by them are called the clause c_j. Clauses are separated from each other by full rows. If a literal x_i appears in the clause c_j, we place a literal gadget in the x_i-column of the clause gadget c_j. If a literal $\bar{x}_i$ appears in the clause c_j, we place a literal gadget in the $\bar{x}_i$-column of the clause gadget c_j. The literal in column x_i of clause c_j is to the left of the literal in column x_i of clause c_h for $j < h$. Similarly, the literal in column $\bar{x}_i$ of clause c_j is to the left of the literal in column $\bar{x}_i$ of clause c_h for $j < h$. So no vertical line stabs the interior of more than one literal.

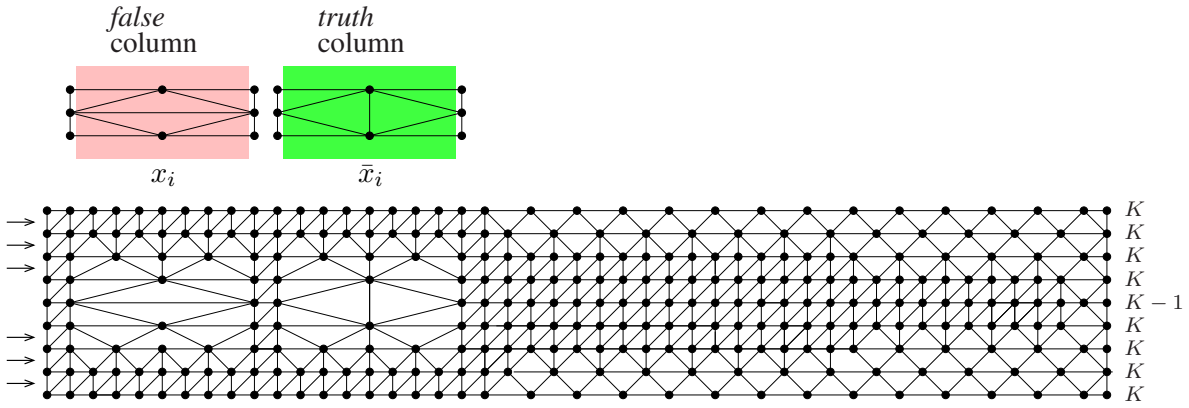

Figure 5: A variable gadget and how it is embedded in a grid of points. Arrows indicate full rows.

By adding points to the right of the literals we ensure that a horizontal line through the top or bottom row of the three literals of a clause has exactly K points, and a horizontal line through the middle horizontal line of the three literals of a clause has exactly $K-2$ points. So two edges along these middle lines may be missing in a triangulation of minimum crossing number, but no more than two. As we will argue later, these edges will be missing in the interior of two of the literals. We call the missing of the horizontal middle edge in a literal the *false* setting of the literal, and the presence of this edge within a literal the *true* setting of the literal.

Finally, in adding points at the bottom of the clauses we ensure the following vertical point counts. First of all, the columns neighboring the variables have to be full; in particular, vertical lines that stab the left or the right points of a variable rectangle have K points each. A vertical line passing through the middle line of a variable rectangle has $K-1$ points. Together with the $K-1$ points on the horizontal middle line of each variable this implies that in any triangulation of minimum crossing number each variable will have exactly one *true* and exactly one *false*-column. A vertical line that passes through a vertical boundary of a literal, but not through a vertical boundary of a variable, receives $K-1$ points. A vertical line through the vertical middle line of some literal has only $K-3$ points. Refer to Figure 4: When the distribution of points is done appropriately, we end up with a configuration P of points that has a rectangular convex hull.

Let $B(x_0, x_1, \ldots, x_{n-1})$ be satisfiable. We show that P has a triangulation of crossing number $2K-1$ that is minimum by Lemma 4. All full rows and full columns are fully triangulated. If variable x_i has the value *true*, we triangulate the interior of the two rectangles of variable x_i in such a way that the x_i-column becomes this variable's *true*-column, and the $\bar{x}_i$-column becomes this variable's *false*-column. The triangulation of the interior of the rectangles is reversed when variable x_i has the value *false*. We set each literal that represents the value *true* to its *true* setting, and set each literal that represents the value *false* to its *false* setting. The triangulation can be completed arbitrarily.

We can now convince ourselves that such a triangulation of P indeed attains *Cr*-$\Delta_2(P) = 2K-1$. No fully triangulated row or column has a crossing number larger than $2K-1$. Because exactly one edge is missing in the horizontal middle line of each respective variable, the two rows of a variable are full. In each clause there is at least one literal in its *true* setting. Therefore, we can afford at most two extra triangles caused by the *false* setting of the other two literals in the clause, and no row of a clause intersects more than $2K-2$ triangles. Finally, we may have one edge missing on the vertical lines passing trough the middle of a literal only in a *true*-column. This condition holds by definition

of the setting of the literals according to the truth value of the variables. The "arbitrary completion" of the triangulation only happens in the lower right of the construction. In this corner, vertical and horizontal lines have a low point count (except for the boundary), and the allowed crossing number is not exceeded.

For seeing the converse, assume that there is a triangulation of P that has crossing number $2K-1$. We show that B is satisfiable. Because a full row or column can be triangulated in such a way that the crossing number of $2K-1$ is not exceeded, we only have to take care of the rows and columns in which we have a degree of freedom, and where the point count is critical by Lemma 4. Because the points in the top and the middle row of a clause add up to $2K-2$, we can afford at most two literals of each clause set to *false*. One literal in each clause has to be set to *true*. The *true* literal has to be in a *true*-column of a variable, for otherwise the vertical crossing number would exceed $2K-1$. Any other literal in this column can also be set to *true*. The horizontal point count of the horizontal lines of the variable forces the second column of this variable to be a *false*-column. In order not to exceed the allowed crossing number in vertical direction, all literals in this column have to be set to *false*. As one easily checks, this yields a consistent setting of the variables, and B is satisfiable.

Finally, the size of our construction is indeed polynomial: Let n and c be the number of variables and clauses of $B(x_0, x_1, \ldots, x_{n-1})$. A rectangle that represents x_i spans two columns for each occurrence of x_i in some clause. So the total width of all variable gadgets is at most $2nc$. Because all variable gadgets are separated by full columns, the total number of columns spanned by the variables is at most $2cn+2n+1$. The number of rows used by the variables and clauses is at most $4n+3c+3$. In order to achieve a point count of K in each line of a full row or column we may have to add points to the right, which requires at most t additional columns, where t is the maximum width of a variable gadget. Therefore K is $O(cn)$. □

3.3 Spanning Trees

The basic construction for showing hardness of finding a spanning tree of minimum stabbing number is very similar to the one for matchings. As before, we use barriers to restrict possible connections: We make use of the arrangement shown in Figure 6.

Lemma 6 *Let S be the arrangement of $3k$ points shown in Figure 6, and let $P \supset S$ have no other points in the horizontal strip indicated by shading. If P has a spanning tree T with stabbing number $k+1$, then no edge of T crosses the shaded region.*

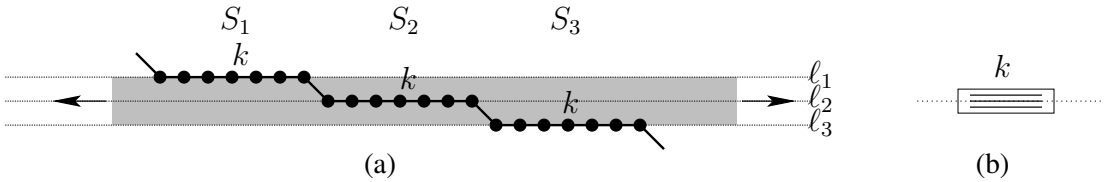

Figure 6: A horizontal barrier gadget: (a) In a spanning tree of stabbing number $k+1$, no line segment may cross the shaded region. (b) Symbol for the barrier gadget; the dotted line indicates the blocked strip.

Proof. Let the three collinear subsets that form S be labeled as S_1, S_2, S_3, as shown in the figure; let ℓ_1, ℓ_2, ℓ_3 be the corresponding horizontal lines. Consider a spanning tree T of P, with $v \in P \setminus S$

lying outside of the strip. Orient all edges of T towards v. Each vertex in S must have outdegree 1, meaning that there are k outgoing edges for each of S_1, S_2, S_3, contributing k to the stabbing numbers along ℓ_1, ℓ_2, ℓ_3. One of the outgoing edges of S_2 must intersect ℓ_1 or ℓ_3 in order to connect S_2 to the rest of the graph; thus, one of those two lines stabs $k+1$ edges, implying the claim. □

Our variable gadgets look as in Figure 7. Note the use of vertical barrier gadgets, and the remaining numbers of segments that may cross the induced dotted lines; the arrows pointing down from the bottom indicate a number of literal gadgets, consisting of 2×2 arrangements of points. See Figure 8 for the resulting overall arrangement.

Lemma 7 *Let S be the arrangement of points shown in Figure 7, with barrier gadgets placed and sized as indicated, and let $P \supseteq S$. Let P be constructed as shown in Figure 8. Then any spanning tree of P that has stabbing number at most $k+1$ must use at least one of the two edges at the bottom of the arrangement, labeled e_t^1 (for* true*) or e_f^1 (for* false*.)*

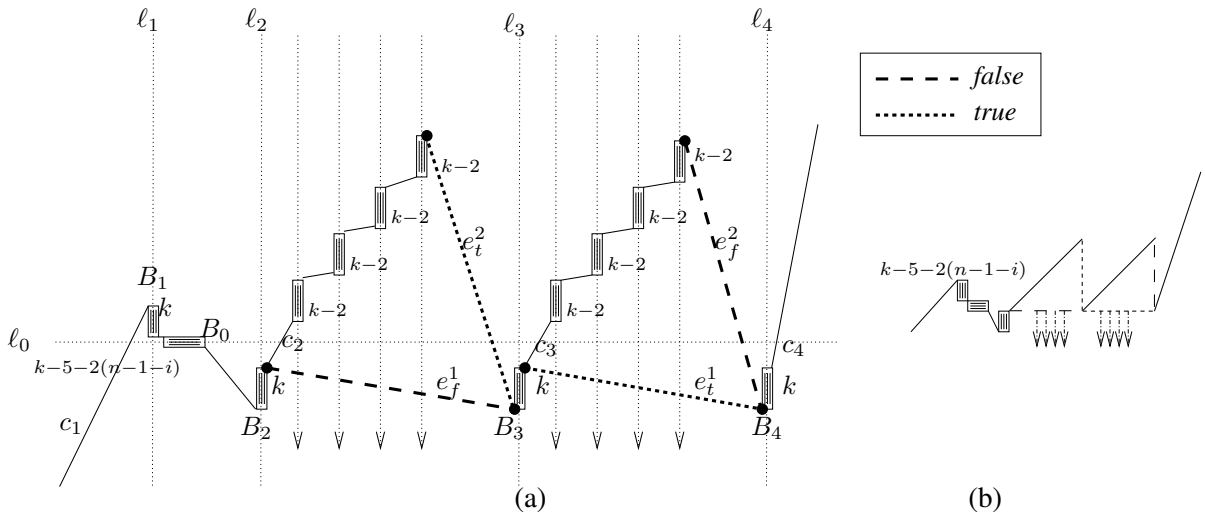

Figure 7: A variable gadget for variable x_i: (a) In a spanning tree of stabbing number $k+1$, the *true* or the *false* setting is chosen. (b) Symbol for the variable gadget.

Proof. Assume there is a spanning tree of stabbing number at most $k+1$. Consider the barrier gadgets labeled B_1, B_2, B_3, B_4, and the corresponding lines, ℓ_1, ℓ_2, ℓ_3, ℓ_4. By the previous lemma, no edge can cross one of those lines. Therefore, the literal boxes below each clause must be connected within the vertical strips bounded by ℓ_2 and ℓ_3, or ℓ_3 and ℓ_4, respectively. This requires at least one edge within each of the two strips to cross the line ℓ_0. Moreover, the lines ℓ_1 and ℓ_4 must not be crossed, implying that the variable gadgets are connected to their neighbors at barriers B_1 and B_4, inducing a stair-like chain of variable gadgets, as shown in Figure 8. Now consider the horizontal barrier B_0, consisting of three groups of $k-5-2(n-1-i)$ points each, where i is the number of the variable, starting with 0. By the previous arguments, the line ℓ_0 has to cross all of the edges connecting the true and false literal boxes of the $n-1-i$ variables with higher numbers, i.e., cross $2(n-1-i)$ edges. Furthermore in variable i there are four other edges that connect points above line l_0 to points below l_0 and so are crossed by l_0. For example these edges could be the ones labeled c_1, c_2, c_3, c_4 in the figure. This allows only one of the edges e_t^2 and e_f^2 to be used for connecting an interior stair to the rest of the stair; thus, at least one "horizontal" line must be used, proving the claim. □

Making use of the above gadgets, we get

Theorem 8 *It is $\mathcal{NP}$-hard to determine St-Tre$_2(P)$.*

Proof. The basic idea for the construction is similar to the one used in the previous sections, making use of Lemmas 6 and 7. The use of gadgets and the overall layout of the construction are shown in Figure 8.

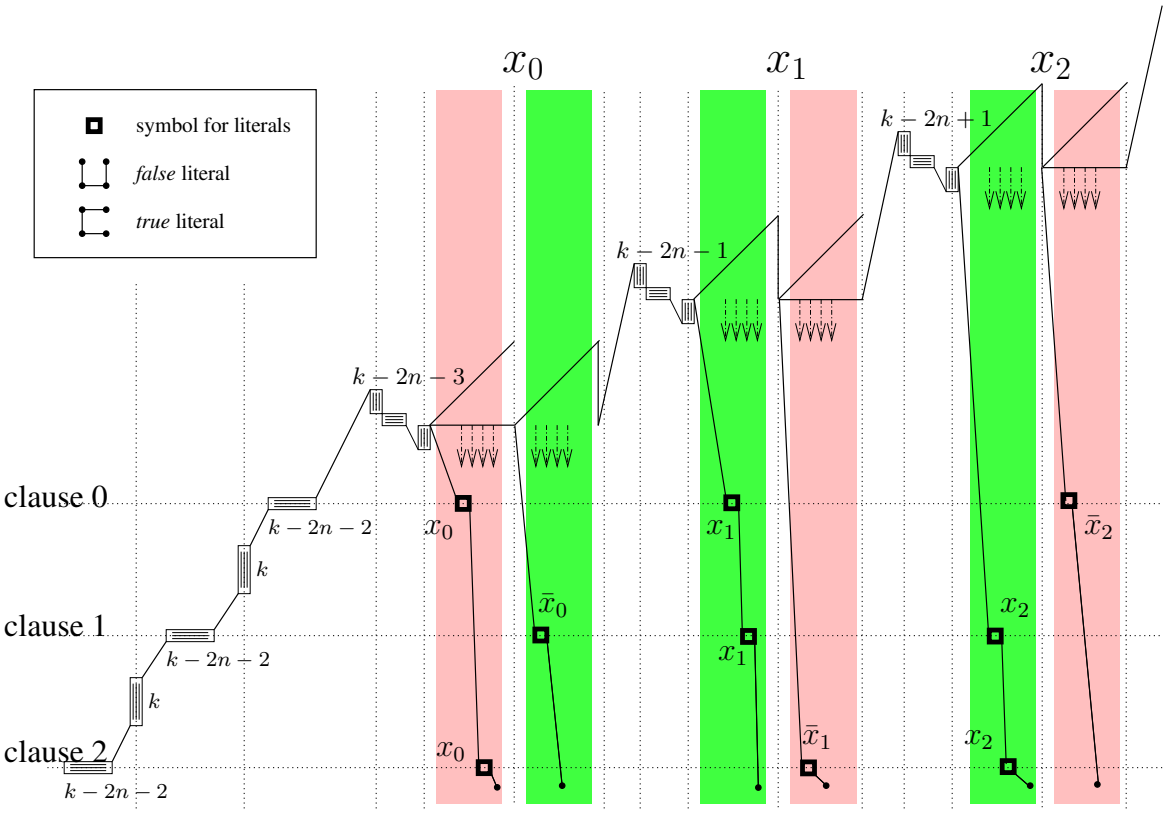

Figure 8: The overall layout for the hardness proof for spanning trees. There is a total of n variables; k is a sufficiently large number, in question is the existence of a spanning tree with stabbing number $k+1$. Shown is the representation of the 3SAT instance $(x_0 \vee x_1 \vee \bar{x}_2) \wedge (\bar{x}_0 \vee x_1 \vee x_2) \wedge (x_0 \vee \bar{x}_1 \vee x_2)$, for $n = 3$, with $x_0 =$ *false* and $x_1 = x_2 =$ *true*.

Given a Boolean expression $B(x_0, x_1, \ldots, x_{n-1})$, we can find a point set that has a spanning tree of a certain stabbing number if and only if B is satisfiable. Consider the point set as given in Figure 8 with $k = 3n$. If $B(x_0, x_1, \ldots, x_{n-1})$ has a truth assignment, we first connect the point on the left side of the drawing into one long path. This path contains all horizontal and vertical barrier gadgets as shown in Figure 8. We connect the variable and literal gadgets according to their values in B. In each *true*- and *false*-column, there is point lower than all horizontal barriers, and to the right of all literals in that column. We connect this point to the left lowest point of the lowest literal in the same column. Because each clause has at least one true literal, the horizontal stabbing number of a stabber through a clause is at most k, because the horizontal barrier on the left contributes $k-2n-1$, the variable(s) set to true and the $n-3$ variables that do not appear in the clause contribute two each, and the variable(s) set to false contribute three each, which is at most $k-2n-1+2(n-2)+6 = k+1$. A horizontal stabber

through the horizontal barrier in the i-th variable gadget stabs $k-5-2(n-1-i)+1$ edges in the barrier, one more edge to the left of the barrier, four edges of the variable to the right of the gadget and two more for each of the $n-i$ variables to its right, for a total of $k-5-2(n-1-i)+1+5+2(n-1-i) = k+1$. A vertical stabber through literals stabs at most $k-2$ edges in the vertical barriers, plus two more, either one from a false literal and one in the variable gadget, or two edges in the true literals. So we have a spanning tree of stabbing number $k+1$.

Conversely, assume that the point set has a spanning tree of stabbing number $k+1$. As we showed in the proof of Lemma 7, the literal gadgets can only be connected within their respective strips, forcing at least $2n$ stabbed edges. Furthermore, each *false* literal (being connected in a "u"-like fashion) causes an additional stabbed edge, while a *true* literal (connected in a "c"-like fashion) does not cause any additional stabbings. Thus, each clause must contain at least one *true* literal. Because of Lemma 7, at least one of the edges e_f^1 and e_t^1 must be present, guaranteeing that only the negated or only the unnegated literals for each variable can be connected in a c-like fashion, i.e., forcing a feasible setting of the variables. Thus, we get a truth setting of the variables that satisfies B. □

As before, this immediately implies

Corollary 9 *It is $\mathcal{NP}$-hard to determine St-Tre(P).*

Proof. Use the construction of Theorem 8, for which the criticality of certain axis-parallel lines requires satisfying a 3SAT instance in order to achieve low stabbing number. Then perturb the position of the gadgets, shifting all points in the same gadget by the same amount, such that no line can intersect the bounding boxes of any three gadgets. This leaves only the axis-parallel lines to be critical, implying the same combinatorial behavior as in the axis-parallel case. □

The hardness proof for minimzing the crossing number has the same structure as the one for stabbing number. Instead of the barrier gadget implies by Lemma 6, we use a slightly different one, as shown in Figure 9.

Lemma 10 *Let S be the $k \times ((k-1)^2+k)$ arrangement of points shown in Figure 9, and let $P \supseteq S$. If P has a spanning tree T with crossing number k, then no other edge of T crosses the shaded region.*

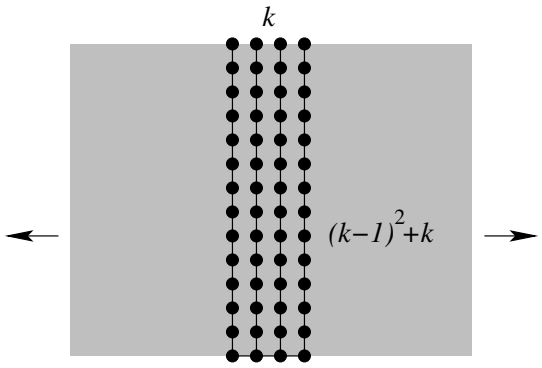

Figure 9: A barrier gadget for showing hardness of minimizing the crossing number of a spanning tree.

Proof. Suppose there is a crossing edge. Consider the $(k-1)^2+k$ "horizontal" sets that consist of k collinear points, and the $(k-1)^2+k$ lines that run through them. Because of the crossing edge, the

intersection of each line with the edges and points within each corresponding horizontal set cannot consist of more than $k-1$ connected components, so it must contain at least one horizontal edge within the set, requiring a total of at least $(k-1)^2+k$ horizontal edges within the arrangement. Now consider the k "vertical" point sets with $(k-1)^2+k$ points each; the k lines through these sets subdivide the plane into $k+1$ vertical strips, $k-1$ of which have width 1. Any of the at least $(k-1)^2+k$ horizontal edges within the arrangement must cross at least one of the bounded strips; thus, the average number of horizontal edges per bounded strip is at least $\frac{(k-1)^2+k}{k-1} = (k-1)+\frac{k}{k-1} > k$. By the pigeonhole principle, this implies that there must be a bounded vertical strip that is crossed by more than k edges, a contradiction to our assumption that the crossing number is at most k. □

As the figure shows, there is a feasible subtree of the gadget, as long as no edge crosses the indicated strip. This allows us to use the arrangement as a barrier gadget. As for the rest of the construction for the proof of Theorem 8, stabbing and crossing numbers coincide, this immediately implies the following.

Theorem 11 *It is $\mathcal{NP}$-hard to determine Cr-Tre$_2(P)$.*

From this is easy to derive the following, again using a perturbation argument.

Corollary 12 *It is $\mathcal{NP}$-hard to determine Cr-Tre(P).*

4 Integer Linear Programs for Minimum Stabbing Number

In view of the negative complexity results for our problems there are two major directions to proceed: providing (good) lower bounds on the minimum stabbing number in order to obtain approximation algorithms; and insisting on optimality despite $\mathcal{NP}$-hardness. Our (integer) linear programming approach is an elegant way to combine both issues. We deal with them in the next two sections.

4.1 Perfect Matchings

In combinatorial optimization one would think of P as the vertex set of a straight-line embedded complete graph $G = (P, E)$. A common representation of a matching M is by its edge incidence vector $x \in \{0,1\}^E$, where $x_{ij} = 1$ if $ij \in M$, and $x_{ij} = 0$ otherwise. Using these variables we are able to state an integer program for finding a perfect matching of minimum stabbing number. For $S \subseteq P$, denote by $\delta(S) = \{ij \in E \mid i \in S, j \notin S\}$ the *cut* induced by S.

$$\text{minimize} \quad k \tag{1}$$

$$\text{s.t.} \quad \sum_{ij \in \delta(i)} x_{ij} = 1 \qquad \forall\, i \in P \tag{2}$$

$$\sum_{ij \in \delta(S)} x_{ij} \geq 1 \qquad \forall\, S \subseteq P,\ |S| \text{ odd} \tag{3}$$

$$\sum_{ij : ij \cap \ell(d) \neq \emptyset} x_{ij} \leq k \qquad \forall \text{ stabbing line } \ell(d) \text{ in direction } d \tag{4}$$

$$x_{ij} \in \{0,1\} \qquad \forall\, ij \in E \tag{5}$$

We obtain the associated linear programming (LP) relaxation by replacing (5) by

$$x_{ij} \geq 0 \tag{5'}$$

The inequalities (2) and (3) are necessarily satisfied for any perfect matching given by x. In his seminal paper Edmonds [8] showed that—together with (5′)—these inequalities already constitute the complete description of the perfect matching polytope, that is, its extreme points exactly correspond to the incidence vectors of perfect matchings in G. In the stabbing constraints (4) we count the number of intersections of matching edges with any given line; this number is bounded by the variable k, and k is minimized. We have to choose this way of modeling because of our min-max objective. An optimal solution x to the integer program (1)–(5) represents a matching with stabbing number exactly *St-Mat*(P).

Note that without loss of generality, it suffices to restrict the set of stabbing constraints (4) to a set of polynomial size: When sweeping a stabbing line over P its stabbing number changes only at a vertex. Therefore, we only need to check a linear number of lines in each direction. For the same reason, "all" directions reduce to the $O(n^2)$ combinatorial directions determined by all pairs of vertices of G. On the other hand, we have exponentially many so-called *blossom inequalities* (3). However, one can check in polynomial time whether a given x violates some blossom inequality, and if so, one such inequality is identified [20]. This polynomial-time *separation* allows us to solve the linear programming relaxation(1)–(5′) in strongly polynomial time [22, Thm. 5.11] by means of the ellipsoid method [18]. An optimal solution x will in general be fractional, and we speak of *fractional stabbing number* in this context. It is a lower bound on *St-Mat*(P).

4.2 Spanning Trees

There are several polynomial-size LP formulations for spanning trees, see e.g., [16]. However, similar to matchings, we choose an exponential-size integer program that is again based on cut constraints. We will exploit their useful properties when we come to approximation algorithms. We directly state the LP relaxation.

$$\text{minimize} \quad k \tag{6}$$

$$\text{s.t.} \quad \sum_{ij \in E} x_{ij} = n - 1 \tag{7}$$

$$\sum_{ij \in \delta(S)} x_{ij} \geq 1 \qquad \forall S \subseteq P \tag{8}$$

$$\sum_{ij : ij \cap \ell(d) \neq \emptyset} x_{ij} \leq k \qquad \forall \text{ stabbing line } \ell(d) \text{ in direction } d \tag{9}$$

$$x \geq 0 \tag{10}$$

Equation (7) ensures the right number of edges in a tree solution, and connectivity is given by (8). Again, violated constraints (8) can be identified with a minimum cut calculation. Thus, this LP can also be solved in polynomial time.

5 Iterated Rounding

We have no hope for a straightforward approximation algorithm by simply rounding up a fractional solution: We use Figure 13 as an indication that there may not be a guaranteed lower bound on the smallest fraction appearing in an optimal fractional solution. However, we choose our linear programs such that some constraints have a *cut structure*: For given families of subsets of vertices we require a certain number of edges leaving such subsets, like in the blossom inequalities. Such a structure is the essence of many network design problems, a very general form of which is the so-called *generalized Steiner network problem*.

Jain [14] introduced a very elegant technique to achieve a 2-approximation algorithm for this problem. Interestingly, it is still based on rounding but does not rely on the smallest occurring fractional values, but on the largest ones. A deep polyhedral argument implies that any fractional solution to an LP formulation of the generalized Steiner network problem must have an edge of value at least $1/2$. Such a *heavy* edge is rounded up and fixed, and the resulting LP, for which the original fractional solution is still feasible, is solved again. The key ingredient is that the modified LP also always has a heavy edge, so the process can be iterated, hence the name: *iterated rounding*.

It is tempting to consider such an approach for deriving approximation algorithms for minimizing the stabbing number of matchings and spanning trees. However, Jain's crucial lemma guaranteeing an edge of value at least $1/2$ does no longer apply in the presence of stabbing constraints: Figure 10 shows a point set for which the optimal fractional axis-parallel stabbing number is achieved by a unique fractional matching with maximum edge weight $1/3$.

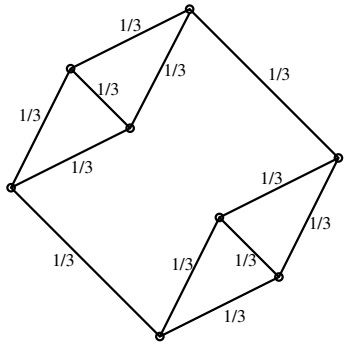

Figure 10: An optimal fractional solution of value 4/3 with maximum edge weight $1/3$.

In the following we show how to make use of the underlying geometry of the problem. The *support graph* of a (fractional) solution x consists of all edges e that have a strictly positive value $x_e > 0$. By showing this graph to be planar, we establish the existence of a heavy edge for optimal fractional solutions to our stabbing LPs. Planarity is proven via an uncrossing 2-exchange argument that for matchings (Lemma 13) requires some extra care because of the blossom inequalities. The proof for spanning trees (Lemma 14) is almost completely analogous.

Lemma 13 *For any even set of points in the plane, there is a fractional perfect matching x of minimum stabbing number, such that the support graph of x is planar. Such a fractional matching can be found in polynomial time.*

Proof. Among the fractional perfect matchings of minimum stabbing number, consider a solution x that minimizes the total stabbing number, i.e., the sum of stabbing numbers over all combinatorial

lines. We claim that the support graph of x cannot contain any crossing pair of edges. Refer to Figure 11.

Suppose there were $e_{13} := \{v_1, v_3\}$, $e_{24} := \{v_2, v_4\}$ with $x(e_{13}) > 0$ and $x(e_{24}) > 0$, such that e_{13} and e_{24} cross. Consider $e_{12} := \{v_1, v_2\}$, $e_{34} := \{v_3, v_4\}$, $e_{14} := \{v_1, v_4\}$, $e_{23} := \{v_2, v_3\}$, and a sufficiently small $0 < \varepsilon < \min\{x_{e_{13}}, x_{e_{24}}\}$. As $\sum_{e \in \delta(v_i)} x_e = 1$ and $x(e_{13}) > \varepsilon$, $x(e_{24}) > \varepsilon$, we have $x(e_{12}) < 1 - \varepsilon$, $x(e_{34}) < 1 - \varepsilon$, $x(e_{14}) < 1 - \varepsilon$, $x(e_{23}) < 1 - \varepsilon$.

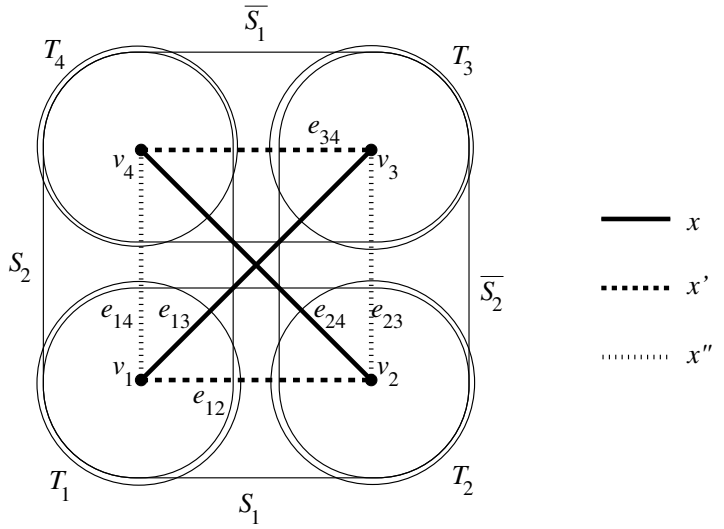

Figure 11: Uncrossing a fractional solution while preserving all blossom inequalities.

Consider two possible alternative solutions arising from "uncrossing" e_{13} and e_{24}. Let x' and x'' be defined by

$$x'_e := \begin{cases} x_e - \varepsilon & \text{for} \quad e \in \{e_{13}, e_{24}\}, \\ x_e + \varepsilon & \text{for} \quad e \in \{e_{12}, e_{34}\} \\ x_e & \text{for} \quad \text{all other } e, \end{cases} \tag{11}$$

and

$$x''_e := \begin{cases} x_e - \varepsilon & \text{for} \quad e \in \{e_{13}, e_{24}\}, \\ x_e + \varepsilon & \text{for} \quad e \in \{e_{14}, e_{23}\}, \\ x_e & \text{for} \quad \text{all other } e. \end{cases} \tag{12}$$

By convexity, both x' and x'' satisfy all stabbing constraints that are valid for x. Furthermore, it is easy to see that both x' and x'' have shorter total stabbing number than x. Thus, it suffices to argue that at least one of these solutions satisfies all blossom inequalities.

Assume that for each of the two alternative solutions, there is a violated blossom inequality, i.e., there are two odd sets, $S_1 \subset P$ and $S_2 \subset P$, such that $\sum_{e \in \delta(S_1)} x'(e) < 1$ and $\sum_{e \in \delta(S_2)} x''(e) < 1$, while $\sum_{e \in \delta(S_1)} x(e) = 1$ and $\sum_{e \in \delta(S_2)} x(e) = 1$. Let $s_i := |\{v_1, v_2, v_3, v_4\} \cap S_i|$. It is straightforward to see that for $s_i \in \{0, 1, 3, 4\}$, both x' and x'' satisfy all blossom inequalities that are valid for x. Therefore, we only need to consider $v_1, v_2 \in S_1$ and $v_3, v_4 \in \overline{S_1}$, and $v_1, v_4 \in S_2$ and $v_2, v_3 \in \overline{S_2}$. Let $T_1 := S_1 \cap S_2$, $T_2 := \overline{S_1} \cap S_2$, $T_3 := S_1 \cap \overline{S_2}$, $T_4 := \overline{S_1} \cap \overline{S_2}$ and $E_{ij} := \{e = \{x, y\} \in E \mid x \in T_i, y \in T_j\}$. As $|T_1 \cup T_2|$, $|T_1 \cup T_4|$, $|T_2 \cup T_3|$, and $|T_3 \cup T_4|$

are all odd, we may assume without loss of generality that $|T_1|$ and $|T_3|$ are odd, and $|T_2|$ and $|T_4|$ are even. Then we have

$$\begin{aligned} 2 &= \sum_{e\in\delta(S_1)} x(e) + \sum_{e\in\delta(S_2)} x(e) \\ &= \sum_{e\in\delta(T_1)} x(e) + \sum_{e\in\delta(T_3)} x(e) + 2\sum_{e\in E_{24}} x(e) \\ &\geq \sum_{e\in\delta(T_1)} x(e) + \sum_{e\in\delta(T_3)} x(e) + 2x(e_{24}) \\ &> \sum_{e\in\delta(T_1)} x(e) + \sum_{e\in\delta(T_3)} x(e). \end{aligned}$$

This implies that $\min\{\sum_{e\in\delta(T_1)} x(e), \sum_{e\in\delta(T_3)} x(e)\} < 1$, contradicting the assumption that x satisfies all blossom inequalities. □

When actually trying to find a noncrossing matching, it suffices to consider an additional term in the objective function of our LP that refers to the total length of the edges. This ensures that the total length of matching edges is minimized, avoiding crossings in the first place.

Lemma 14 *For any set of points in the plane, there is a fractional spanning tree x of minimum stabbing number, such that the support graph of x is planar. Such a fractional spanning tree can be found in polynomial time.*

Proof. We proceed completely analogous to the proof of the previous lemma to deduce that we can perform an uncrossing 2-exchange. In addition, note that constraints (8) imply that in the presence of two crossing edges in the support graph, no two adjacent ones of the other four edges $\{v_1, v_2\}$, $\{v_2, v_3\}$, $\{v_3, v_4\}$, $\{v_4, v_1\}$ may both have weight 1. □

Theorem 15 *For any even set of points in the plane, there is a fractional perfect matching x of minimum stabbing number that has an edge of weight at least $1/5$. For any set of points in the plane, there is a fractional spanning tree x of minimum stabbing number that has an edge of weight more than $1/3$.*

Proof. For both problems, consider a fractional vertex with a planar support graph. To see the claim for matchings, note that there must be a vertex with degree at most five; as the total weight for each vertex is 1, the claim follows. To see the claim for spanning trees, note that the total edge weight is $n-1$, and the number of edges is at most $3n-6$, implying that the average weight is larger than $1/3$. □

Theorem 15 provides the basic ingredient for an iterated rounding approach: At each iteration, fix the weight of an edge of maximum fractional weight to one, and re-solve the linear program. In each iteration, the number of edges with fractional weight is reduced, so we get an overall polynomial-time method for finding an integral solution.

Unfortunately, Jain's original proof only guarantees a constant-factor approximation for objective functions that arise as the (weighted) sum of the edge variables. However, the situation is different for our objective function which is a *maximum* over certain sums of edge variables, so an additional argument is needed for establishing a constant-factor guarantee. We are hopeful that this argument can be completed some time in the future [15]. As we show in the following Section 6, the practical performance seems to be even better than the theoretically possible guarantees of 5 and 3.

6 Computational Results

6.1 Experimental Setup

We compiled a test suite of various instances on which we evaluated our linear/integer program (1)–(5) and the iterated rounding technique for *St-Mat*$_2(P)$. The suite includes ten instances with up to 442 points from the TSPLIB [21] (last point removed from odd cardinality instances; therefore the results reported here are more meaningful than those in [10]); the C-class ("clustered") of Solomon's instances of the vehicle routing problem [24] with 100 points each; 25 regular grids with 20 to 360 points, based on grids of size 5×5 up to 20×20 in which 20% of the points are removed (chosen uniformly at random); and a set of instances with up to 100 random points in the plane. Even though we experimented with available separation routines for blossom inequalities these are not included in the LPs on which we report below, as solution quality is already excellent.

Tables 1 and 2 display our results on a 2.8GHz Pentium 4 Linux PC with 1GB main memory, using the commercial LP/IP solver CPLEX 9.1. For each instance we list its name, which indicates the number of points; this number is reduced by one for odd names to allow a perfect matching. Also listed are the optimal objective function values for the linear (LPopt) and the integer (IPopt) program, together with the respective CPU time in seconds. The last column displays the approximate stabbing number obtained from iterated rounding. For solving the integer programs we set a time limit of four CPU hours which was exceeded for some large instances. This is indicated by brackets around the corresponding value. In that case we report the CPU seconds it takes to obtain the listed best known solution. To provide some intuition what fractional matchings of minimum (fractional) stabbing number look like, we show several of them in Figures 12–15. The edge weight is proportional to the thickness of edges in the drawing.

6.2 Brief (Additional) Observations

In fractional solutions variables may assume rather arbitrary fractional and small values; this is also true when blossom inequalities are added. The colinearity of points in the grid instances enables us to reduce the number of stabbing constraints, resulting in significantly reduced computation times. The clustering of points in the vehicle routing instances obviously facilitate the LP/IP solution process, as was to be expected. However, this observation is interesting in practice where the data is usually well structured, as opposed to randomly distributed.

In our experiments, the stabbing number obtained from iterated rounding is extremely close to the optimum: it is never off by more than by an *absolute value* of one or two, i.e., much better than predicted by our analysis (Lemma 13). Computation times are comparable to solving the linear program because an LP solver will exploit the fact that linear programs only differ very slightly from iteration to iteration, and will perform a "warm start." We also experimented with a "one good shot at once" approach that is based on the fact that each fractional matching is the convex combination of perfect matchings, by finding a maximum weight perfect matching in the support graph of the LP solution. This tends to give very good feasible solutions and certainly deserves further evaluation, both from a computational and from a theoretical point of view.

We also made an experiment (reported in [10]) to show that the stabbing constraints seem to completely destroy the polyhedral structure of the matching polytope. Half of the original TSPLIB instances we used are infeasible (because they originally had an odd number of points), and this is not

Instance	LP opt	LP CPU	IP opt	IP CPU	iterated
ulysses22	1.992	0.00	2	0.01	2
berlin52	2.815	0.02	4	0.90	5
lin105	5.500	0.15	6	80.57	8
bier127	4.297	0.34	(6)	(3.90)	7
u159	15.000	0.15	15	2.37	15
ts225	13.700	0.35	(15)	(122.28)	16
tsp225	11.500	0.32	12	7.66	12
a280	10.500	4.26	(12)	(284.80)	12
lin318	8.113	12.65	(10)	(6825.48)	11
pcb442	16.500	20.71	17	3289.41	18
c101	7.000	0.03	7	0.54	8
c102	7.000	0.03	7	0.54	8
c103	7.000	0.03	7	0.53	8
c104	7.000	0.03	7	0.53	8
c105	7.000	0.03	7	0.53	8
c106	7.000	0.04	7	0.54	8
c107	7.000	0.03	7	0.54	8
c108	7.000	0.03	7	0.53	8
c201	6.000	0.03	6	2.37	7
c202	6.000	0.03	6	2.37	7
c203	6.000	0.03	6	2.36	7
c204	6.000	0.03	6	2.37	7
c205	6.000	0.04	6	2.35	7
c206	6.000	0.04	6	2.37	7
c207	6.000	0.04	6	2.46	7
c208	6.000	0.40	6	4.18	7

Table 1: TSPLIB and clustered instances: Comparison of fractional and integer optimal stabbing number *St-Mat*$_2(P)$, and the one obtained from iterated rounding. Brackets around values indicate an exceeded time limit of four CPU hours, and we report the best known solution obtained after the time given.

Instance	LP opt	LP CPU	IP opt	IP CPU	iterated
grid5a	2.500	0.00	3	0.01	3
grid5b	2.750	0.00	3	0.00	3
grid5c	2.750	0.01	3	0.01	4
grid5d	2.000	0.00	3	0.01	3
grid5e	2.500	0.00	3	0.01	3
grid8a	5.003	0.01	6	0.03	6
grid8b	5.125	0.01	6	0.05	6
grid8c	5.000	0.00	5	0.05	6
grid8d	5.429	0.01	6	0.04	7
grid8e	5.403	0.00	6	0.21	6
grid10a	4.250	0.01	5	0.17	6
grid10b	4.250	0.01	5	0.13	6
grid10c	5.250	0.01	6	0.19	6
grid10d	4.500	0.02	5	1.17	6
grid10e	5.000	0.01	5	0.32	6
grid15a	6.000	0.03	6	15.92	7
grid15b	7.500	0.03	8	1.11	8
grid15c	6.000	0.03	(7)	(7.01)	7
grid15d	6.500	0.03	7	54.73	7
grid15e	6.750	0.02	7	761.33	7
grid20a	9.167	0.98	(11)	(43.17)	12
grid20b	9.250	0.27	(11)	(84.97)	11
grid20c	9.500	1.54	(11)	(33.05)	12
grid20d	9.500	2.24	(11)	(448.00)	12
grid20e	10.000	2.89	11	1169.66	12
random10a	1.750	0.00	2	0.01	2
random10b	1.834	0.00	2	0.00	2
random10c	1.750	0.00	2	0.01	2
random10d	1.700	0.00	2	0.01	2
random10e	1.813	0.00	2	0.01	2
random50a	2.595	0.25	3	19.83	4
random50b	2.628	0.23	3	1.91	4
random50c	2.669	0.23	4	30.77	4
random50d	2.662	0.22	4	15.99	4
random50e	2.790	0.33	4	25.54	4
random100a	3.376	5.57	(5)	(14.00)	6
random100b	3.406	1.04	(5)	(13.81)	5
random100c	3.247	0.99	(5)	(16.31)	6
random100d	3.211	0.89	(5)	(7.35)	6
random100e	3.233	0.90	(5)	(5.84)	5

Table 2: Results for grids and random instances

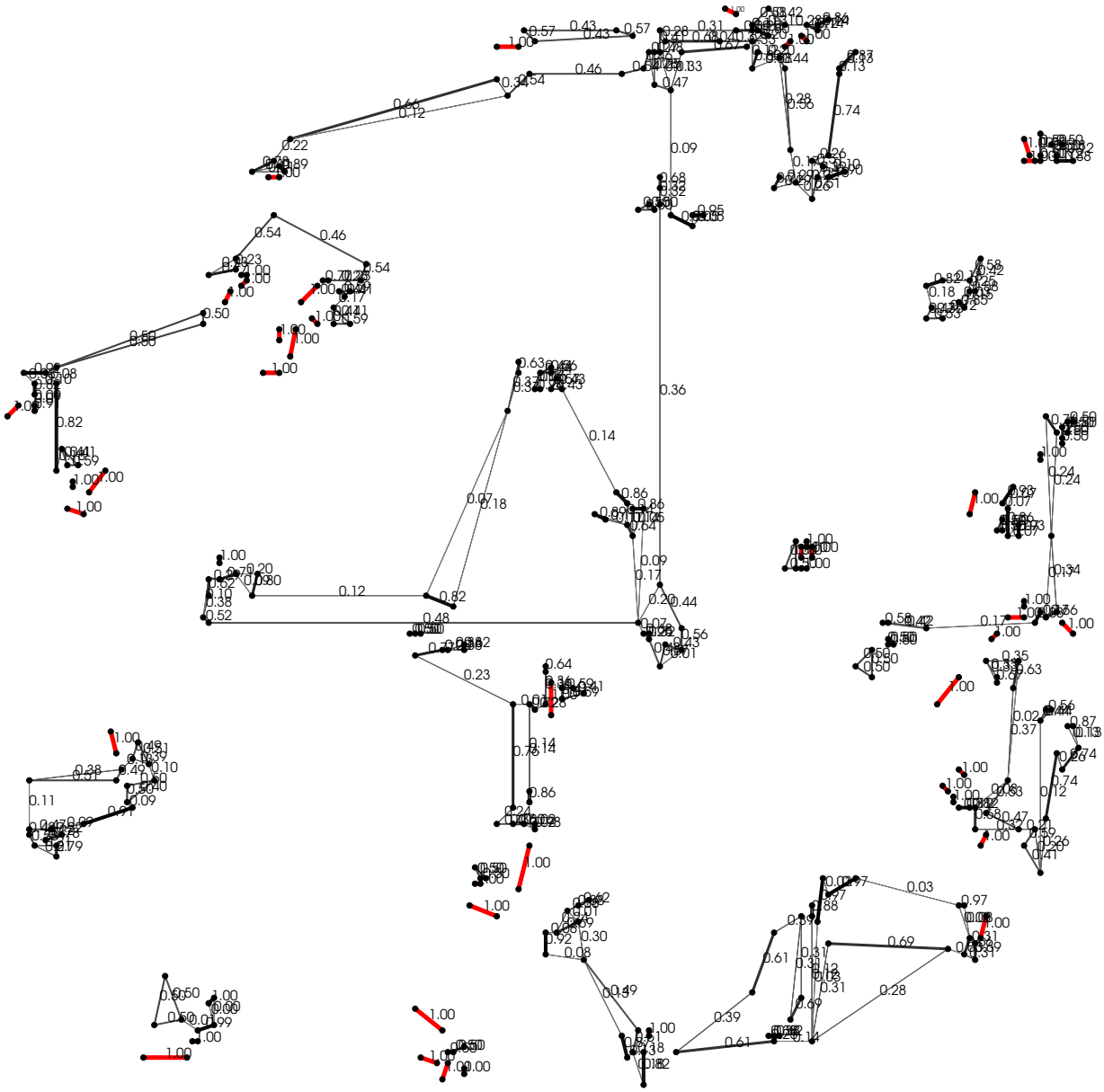

Figure 12: LP optimal solution for a “clustered” instance similar to c101, but with 400 points

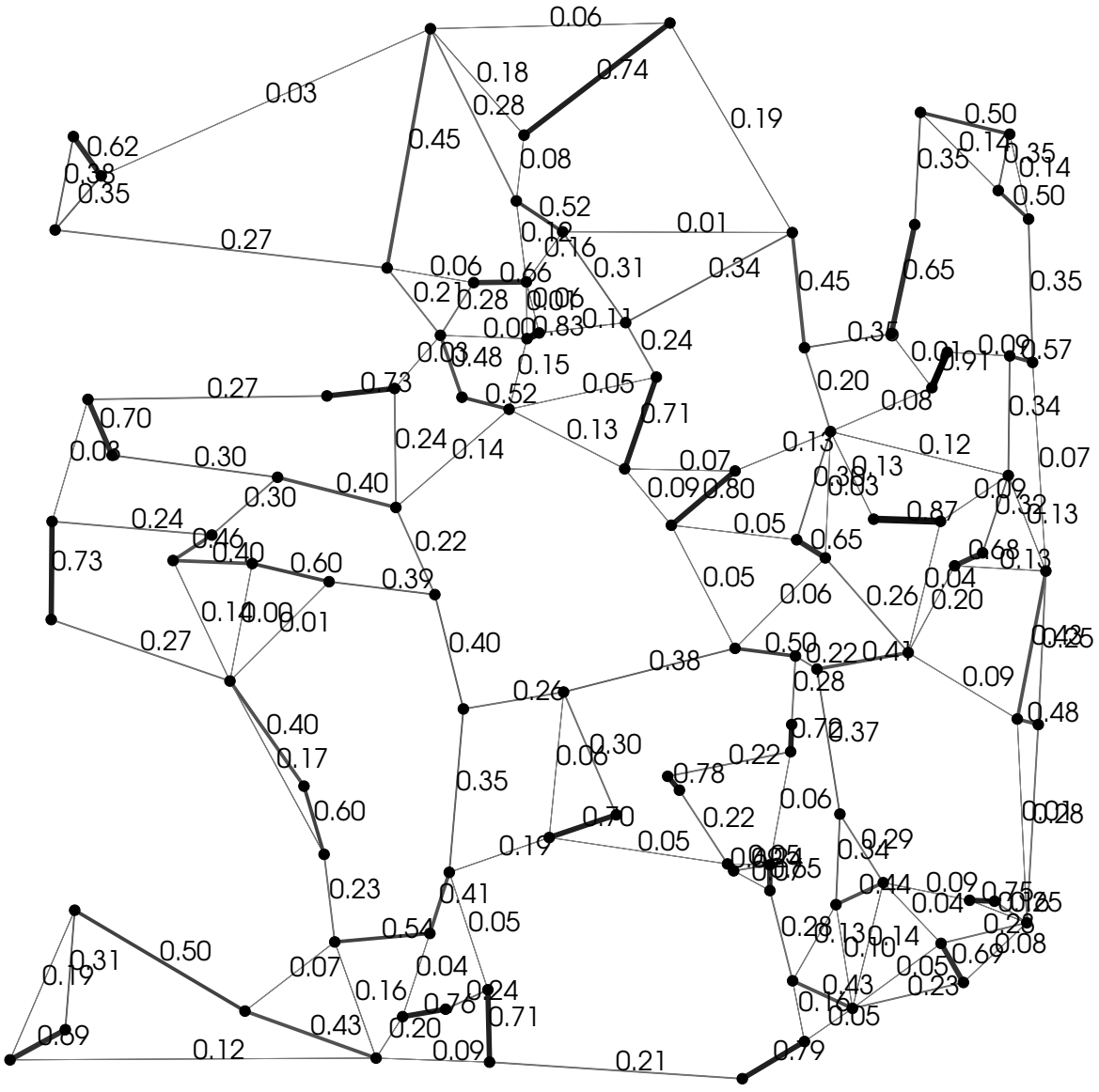

Figure 13: LP optimal solution for random100a

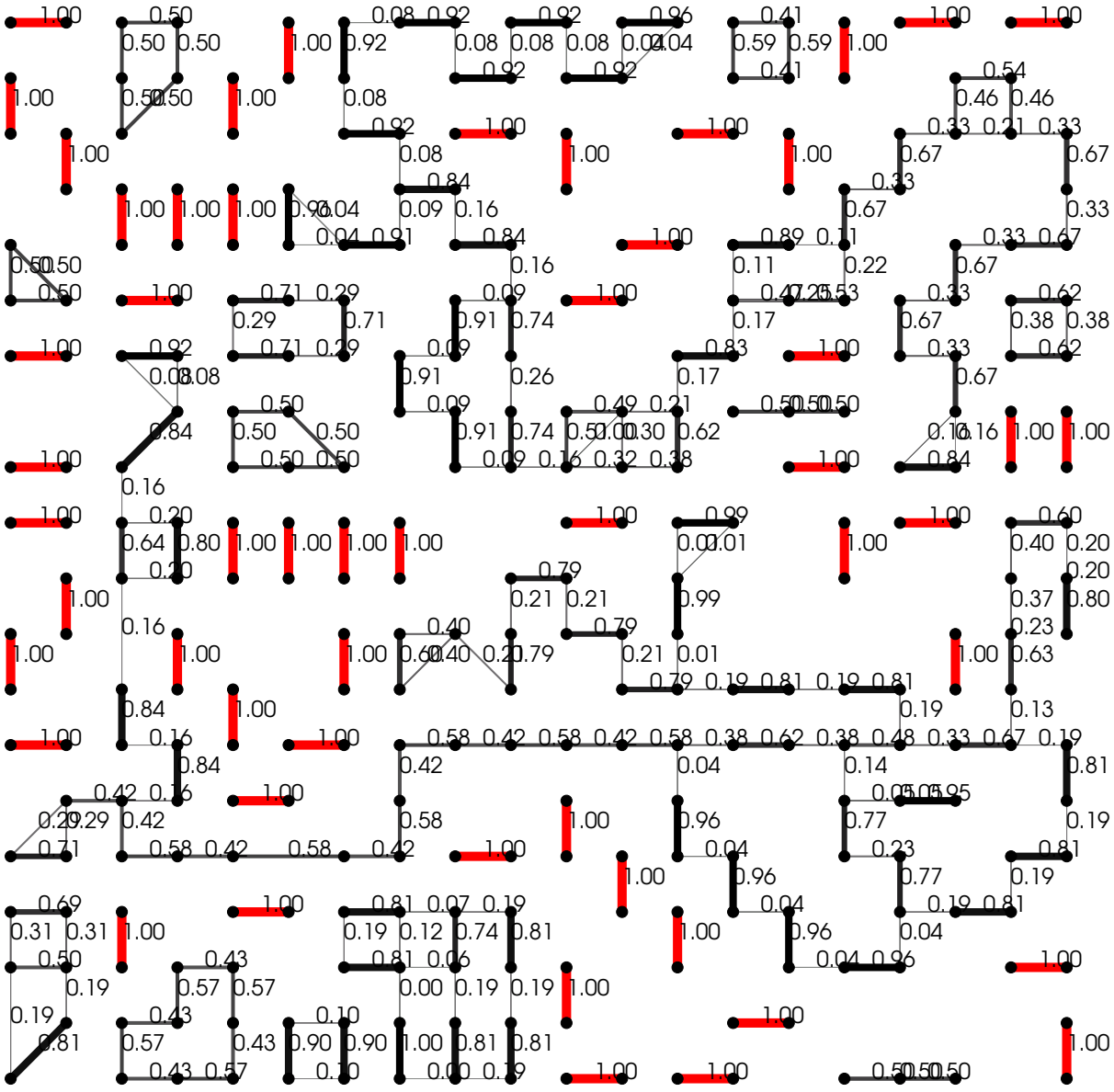

Figure 14: LP optimal solution for grid20b

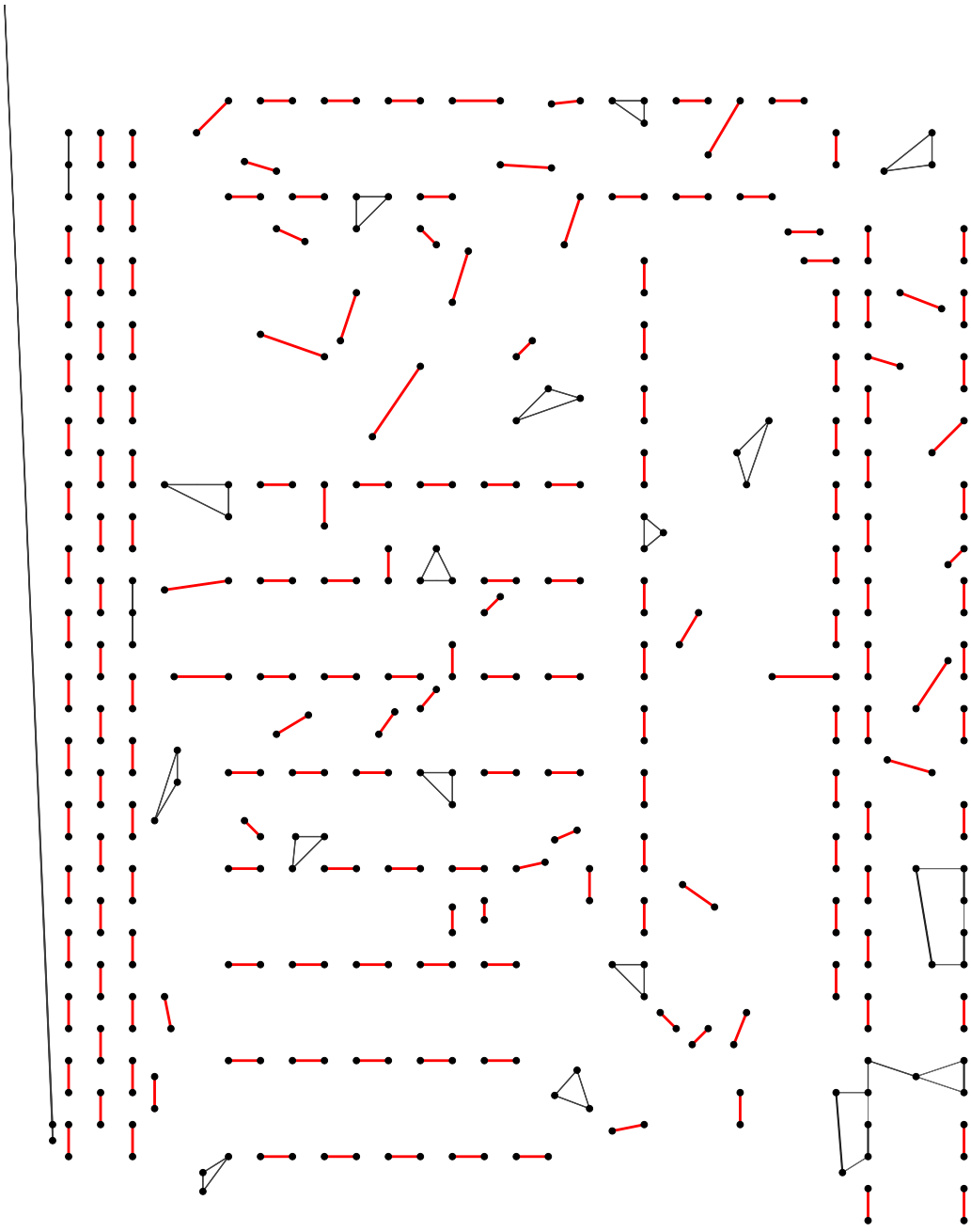

Figure 15: LP optimal solution for pcb442; ‘pcb’ stands for ‘printed circuit board’

detected by the state-of-the-art CPLEX IP solver within four CPU hours. Iterated rounding terminates (quickly) in this case with a non-perfect matching with one point unmatched.

Feasible integer solutions of good quality are usually obtained rather quickly by our integer programs. The time-consuming part appears to be a proof of optimality, but the lower bound increases only slowly in the branch-and-bound tree. It would be worthwhile to investigate strengthening the lower bound obtained from the LP relaxation by means of valid inequalities ("cutting planes").

7 Notes and Conclusion

We have presented the first algorithmic paper on stabbing numbers, resolving the long-standing open question of complexity, and providing an approach that appears to be useful in theory and in practice. There are a number of interesting open questions.

We were not able to extend our $\mathcal{NP}$-hardness proof to the case of finding a triangulation of minimum (general) stabbing number. Our proofs rely on a strong degeneracy of the point set, and it would be interesting to see a proof for points in general position.

Probably the most intriguing open question spawned by our work is whether the iterated rounding scheme suggested by the existence of a heavy edge in an optimal fractional solution to our linear programs (Lemmas 13 and 14) does indeed lead to a constant-factor approximation algorithm. Also, the use of the Ellipsoid method (at least as a theoretical argument) is not "combinatorial", which always has to be considered a drawback.

Another interesting question is to decide the existence of structures of small constant stabbing number. As the hardness proof for deciding the existence of a matching of stabbing number 5 illustrates, this is still not an easy task. From some solvable special cases, we only note one:

Theorem 16 *St-Tre$_2(P)$=2 and St-Mat(P)=2 can be decided in polynomial time.*

One may also ask for minimizing the *average* instead of the maximum stabbing number, and refer to the average over the whole continuum of lines intersecting a set of line segments, instead of just a combinatorial set of representatives. This, however, amounts to solving problems of minimum length, with all implications to hardness and approximation.

Theorem 17 *A set of line segments has minimum average (axis-parallel, resp.) stabbing number, iff the overall Euclidean (Manhattan, resp.) length of all line segments is minimum.*

We remark that a linear program for minimizing the average stabbing number can be written with a sum in the objective function (instead of a maximum as we had to model it), allowing to directly apply our iterated rounding technique and obtaining the desired approximation factors of 3 and 5, respectively.

Acknowledgments

We thank Joe Mitchell for pushing us to work on this problem, and repeated discussions that assured further progress. We also thank Kamal Jain for some discussions on iterated rounding.

References

[1] P.K. Agarwal. Ray shooting and other applications of spanning trees with low stabbing number. *SIAM J. Computing*, 21(3):540–570, 1992.

[2] P.K. Agarwal, B. Aronov, and S. Suri. Stabbing triangulations by lines in 3D. In *Proc. 11th ACM Sympos. Computational Geometry*, pages 267–276, 1995.

[3] E.A. Arkin, M.E. Bender, E.D. Demaine, S.P. Fekete, J.S.B. Mitchell, and S. Sethia. Optimal covering tours with turn costs. In *Proc. 12th ACM-SIAM Sympos. Discrete Algorithms*, pages 138–147, 2001. Full version to appear in *SIAM Journal on Computing*.

[4] B. Aronov, H. Brönnimann, A.Y. Chang, and Y.-J. CHiang. Cost-driven octree construction schemes: an experimental study. *Comput. Geom. Theory Appl.*, 31:127–148, 2005.

[5] B. Aronov and S. Fortune. Approximating minimum-weight triangulations in three dimensions. *Discrete Comput. Geom.*, 21(4):527–549, 1999.

[6] M. de Berg and M. van Kreveld. Rectilinear decompositions with low stabbing number. *Inform. Process. Lett.*, 52(4):215–221, 1994.

[7] E.D. Demaine, J.S.B. Mitchell, and J. O'Rourke. The open problems project. `http://cs.smith.edu/~orourke/TOPP/Welcome.html`, 2003.

[8] J. Edmonds. Maximum matching and a polyhedron with 0,1-vertices. *J. Res. Nat. Bur. Standards*, 69B:125–130, 1965.

[9] S. P. Fekete. On simple polygonalizations with optimal area. *Discrete Comput. Geom.*, 23:73–110, 2000.

[10] S. P. Fekete, M. E. Lübbecke, and H. Meijer. Minimizing the stabbing number of matchings, spanning trees, and triangulations. In *Proc. 15th ACM-SIAM Sympos. Discrete Algorithms*, pages 430–439, 2004.

[11] M.R. Garey and D.S. Johnson. *Computers and Intractability—A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, San Francisco, 1979.

[12] M. Held, J.T. Klosowski, and J.S.B. Mitchell. Evaluation of collision detection methods for virtual reality fly-throughs. In *Proc. 7th Canadian Conf. Computational Geometry*, pages 205–210, 1995.

[13] J. Hershberger and S. Suri. A pedestrian approach to ray shooting: Shoot a ray, take a walk. *J. Algorithms*, 18:403–431, 1995.

[14] K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[15] K. Jain. Personal communication, 2003.

[16] T.L. Magnanti and L.A. Wolsey. Optimal trees. In M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser, editors, *Network Models*, volume 7 of *Handbooks in Operations Research and Management Science*, chapter 9, pages 503–616. North-Holland, 1995.

[17] J. Matoušek. Spanning trees with low crossing number. *Inform. Theor. Appl.*, 25:102–123, 1991.

[18] M.Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, Berlin, 1988.

[19] J.S.B. Mitchell and J. O'Rourke. Computational geometry column 42. *Int. J. Comput. Geom. Appl.*, 11(5):573–582, 2001.

[20] M. Padberg and M.R. Rao. Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, 7:67–80, 1982.

[21] G. Reinelt. TSPLIB – A traveling salesman problem library. *ORSA J. Comput.*, 3(4):376–384, 1991.

[22] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, Berlin, 2003.

[23] J.R. Shewchuk. Stabbing Delaunay tetrahedralizations. *Discrete Comput. Geom.*, 32(3):339–343, 2004.

[24] M.M. Solomon. VRPTW benchmark problems. `http://w.cba.neu.edu/~msolomon/problems.htm`, 2003.

[25] C. Tóth. Orthogonal subdivisions with low stabbing numbers. In *Proc. 9th International Workshop on Algorithms and Data Structures (WADS 2005)*, volume 3608 of *Springer LNCS*, pages 256–268, 2005.

[26] E. Welzl. On spanning trees with low crossing numbers. In B. Monien and Th. Ottmann, editors, *Data Structures and Efficient Algorithms*, volume 594 of *Springer LNCS*, Berlin, 1992.