Forschungsschwerpunkt S92

Industrial Geometry

http://www.ig.jku.at

INDUSTRIAL GEOMETRY

FSP Report No. 45

Connecting Colored Point Sets

O. Aichholzer, F. Aurenhammer, T. Hackl
and C. Huemer

November 2006

FWF
Der Wissenschaftsfonds.

TU Graz

Connecting Colored Point Sets*

Oswin Aichholzer[†] Franz Aurenhammer[‡] Thomas Hackl[§] Clemens Huemer[¶]

Abstract

We study the following Ramsey-type problem. Let $S = B \cup R$ be a two-colored set of n points in the plane. We show how to construct, in $O(n \log n)$ time, a crossing-free spanning tree $T(B)$ for B, and a crossing-free spanning tree $T(R)$ for R, such that both the number of crossings between $T(B)$ and $T(R)$ and the diameters of $T(B)$ and $T(R)$ are kept small. The algorithm is conceptually simple and is implementable without using any non-trivial data structure. This improves over a previous method in Tokunaga [16] that is less efficient in implementation and does not guarantee a diameter bound. Implicit to our approach is a new proof for the result in [16] on the minimum number of crossings between $T(B)$ and $T(R)$.

1 Introduction

Let S be a set of n points in general position in the plane. Consider an arbitrary two-coloring of S, that is, S is the disjoint union of B and R such that each point in B is colored blue and each point in R is colored red. This two-coloring induces an edge coloring of the complete geometric straight-line graph $K(S)$ spanned by S, in the following way: Edges that connect two points of the same color are given this color, and all other edges of $K(S)$ are given a third color, say green. Finding monochromatic subgraphs of $K(S)$ with special properties is a topic of classical Ramsey theory. For example, there always exists a crossing-free perfect matching that uses only green edges, provided $|R| = |B|$; see [14]. Or, for every (not necessarily induced) two-coloring of the edges of $K(S)$ there exists a monochromatic and crossing-free spanning tree of S; see [12]. By a *crossing-free* geometric graph G we mean that no two edges of G cross, i.e., intersect in a single point that is not a vertex of G. Two geometric graphs G_1 and G_2 are called *compatible* if no edge of G_1 crosses any edge of G_2.

In this paper, we study the question of how to compute efficiently a crossing-free blue spanning tree $T(B)$ for B, and a crossing-free red spanning tree $T(R)$ for R, with particular properties. The

*Research supported by the FWF Joint Research Project 'Industrial Geometry' S9205-N12, Project MCYT-FEDER BFM2003-00368, Project Gen. Cat 2001SGR00224, and Acción Intergrada España-Austria HU2002-0010.

†Institute for Software Technology, University of Technology, Graz, Austria. e-mail: oaich@ist.tugraz.at

‡Institute for Theoretical Computer Science, University of Technology, Graz, Austria. e-mail: auren@igi.tugraz.at

§Institute for Software Technology, University of Technology, Graz, Austria. e-mail: thackl@ist.tugraz.at. Research was done while this author was with the Institute for Theoretical Computer Science, University of Technology, Graz, Austria.

¶Dept. Mat. Aplicada II, Universitat Politecnica de Catalunya, Barcelona, Spain. e-mail: huemer.clemens@upc.es

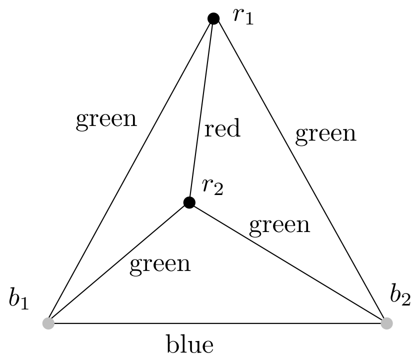

Figure 1: Splitting a blue triangle

objective is to keep small the diameters of the trees $T(B)$ and $T(R)$, as well as the number of crossings between $T(B)$ and $T(R)$.

For arbitrary choices of $T(B)$ and $T(R)$, the red tree clearly may cross the blue tree $\Theta(n^2)$ times, for $n = |S|$. A result in [1] implies that $T(R)$ may be chosen such that each of its edges crosses $T(B)$ only $O(\sqrt{n})$ times, giving an upper bound of $O(n\sqrt{n})$ crossings. Interestingly, this bound can be reduced to $O(n)$. Tokunaga [16] proved that the minimum number of crossings between $T(B)$ and $T(R)$ is exactly $\max\{\frac{g}{2} - 1, 0\}$, where g denotes the number of green edges on the convex hull of S. (Note that g is an even number.) As a corollary, $T(B)$ and $T(R)$ can be chosen to be compatible if and only if $g = 0$ or $g = 2$.

The proof in [16] is constructive, and a straightforward implementation runs in time polynomial in n. However, a sub-quadratic implementation requires complex data structures for dynamic convex hull maintenance [15]. We shall give an alternative construction, that is conceptually simpler and that translates to an $O(n \log n)$ time algorithm without using any non-trivial data structure. Moreover, the constructed spanning trees have their diameters bounded by $O(h + \log n)$ and $O(h \log \frac{n}{h})$, respectively, where h denotes the number of points on the convex hull of S. (We may choose which tree meets the smaller diameter bound.) This is not far from the optimum as a diameter of $\Theta(h)$ may not be avoidable when crossings are kept to a minimum.

In Figures 1 through 7, red (blue) points are shown as black (grey) dots.

2 The triangle case

We first solve a simpler problem, which nevertheless will turn out to be the core of the general problem. Let S be a two-colored point set. A triangle $\triangle$ spanned by three points in S is called a *red* triangle if exactly one edge of $\triangle$ is red, and a *blue* triangle if exactly one edge of $\triangle$ is blue. Notice that each red or blue triangle has exactly two green edges. Throughout, we denote with $CH(S)$ the convex hull of S.

Lemma 1 *Let $S = B \cup R$ be a two-colored set of n points such that $CH(S)$ is a triangle which is either blue or red. There exists a triangulation of S that contains a (blue) spanning tree for B, a (red) spanning tree for R, and a green spanning tree for S.*

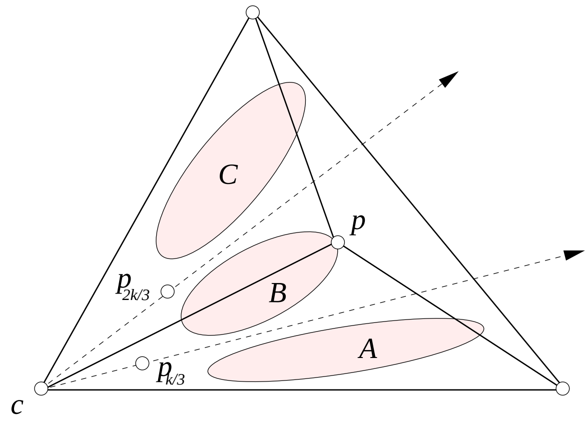

Figure 2: Balanced splitting

Proof. Let $\triangle = CH(S)$. Assume that $\triangle$ is a blue triangle, i.e., $\triangle$ has a single blue edge b_1b_2 and a single red vertex r_1. (The other case is symmetric by exchanging blue and red.) We proceed by induction on the number of points of S in the interior of $\triangle$. Let r interior points be red and b interior points be blue. Clearly, if $r + b = 0$ then the lemma is trivial.

If $r = 0$ and $b > 0$ then we connect r_1 to each blue interior point of $\triangle$ and obtain a fan of green edges. Also, we connect these blue points in their fan order and obtain a blue path, which we finally connect to one of b_1 or b_2. The resulting graph can then be completed to a triangulation of $\triangle$ which contains the desired three spanning trees.

If $r > 0$ then we choose a red point r_2 in the interior of $\triangle$ and connect r_2 to r_1 with a red edge, and also to b_1 and b_2 with two green edges. This splits $\triangle$ into three triangles with fewer interior points. Each of these triangles has exactly two green edges and therefore is either red or blue; see Figure 1. So, by induction, within these triangles the postulated triangulations do exist. Their concatenation gives a triangulation of $\triangle$ that contains a spanning tree for R because of the edge r_1r_2, a spanning tree for B because of the edge b_1b_2, and a green spanning tree for S because of the common point r_2. □

The proof of Lemma 1 is constructive and allows for an efficient implementation. We show that a runtime of $O(n \log n)$ can be achieved by splitting triangles in a balanced way. To this end, the partitioning result in Lemma 2 below is utilized. A similar approach has been used in [4] for solving a different problem on triangulations. More general versions of the lemma (with more involved proofs) have been given in [5, 2], who also showed that the factor of $\frac{2}{3}$ in the lemma is worst-case optimal.

Lemma 2 *Let I be a set of k points in the interior of a triangle $\triangle$. There exists a point $p \in I$ such that the three triangles, obtained by splitting $\triangle$ with the edges from p to the vertices of $\triangle$, contain at most $\frac{2}{3}k$ points of I each. Moreover, p can be computed in time $O(k)$.*

Proof. Let c be a fixed vertex of $\triangle$. With respect to c, consider the cyclic order $p_1, \ldots, p_k$ of the point set I. The point $p_{\frac{k}{3}}$ and the point $p_{\frac{2k}{3}}$ can be computed using an $O(k)$ time median algorithm. (Fractions are rounded appropriately.) The two rays defined by $cp_{\frac{k}{3}}$ and $cp_{\frac{2k}{3}}$, respectively, partition I into three subsets A, B, and C of sizes at most $\frac{k}{3}$; see Figure 2. For splitting $\triangle$, we choose as p a point in B with minimal distance from the line through the edge of $\triangle$ opposite to c. The resulting three triangles contain at most $\frac{2k}{3}$ points of I, because the two triangles that have c as a vertex do not contain points from either A or C, and the third triangle does not contain points from $B \setminus \{p\}$. Clearly, B, and thus p, can be found in $O(k)$ time by checking each point in I. □

Lemma 3 *The three spanning trees in Lemma 1 can be computed in $O(n \log n)$ time.*

Proof. Let S be a point set as in Lemma 1, with $|S| = n$. Then $CH(S)$ is either a blue or a red triangle, $\triangle_0$. Denote with R_0 and B_0 the set of red and of blue points of S, respectively, in the interior of $\triangle_0$. Let $r_0 = |R_0|$ and $b_0 = |B_0|$, and assume $r_0 > 0$ and $b_0 > 0$ first. If $\triangle_0$ is blue then we utilize Lemma 2 to find a point $p \in R_0$ that splits $\triangle_0$ into three triangles $\triangle_1$, $\triangle_2$, and $\triangle_3$, each containing at most $\frac{2r_0}{3}$ points of R_0. Similarly, if $\triangle_0$ is red then we choose $p \in B_0$ such that $\triangle_0$ splits into three triangles $\triangle_1$, $\triangle_2$, and $\triangle_3$ each containing at most $\frac{2b_0}{3}$ points of B_0. Next, R_0 and B_0 are each divided into three subsets R_i and B_i $(i = 1, 2, 3)$, respectively, according to their containment in $\triangle_1$, $\triangle_2$, and $\triangle_3$. These steps take time $O(r_0 + b_0)$ in total, including the time for constructing the red, blue, and green spanning trees so far.

We recur on each $\triangle_i$ until either $r_i = |R_i| = 0$ and $\triangle_i$ is blue, or $b_i = |B_i| = 0$ and $\triangle_i$ is red. This process can be represented by a ternary tree T whose nodes are triangles $\triangle_j$, characterized by the sets (R_j, B_j) they contain. The root of T is (R_0, B_0), and for each level of T, the total number of points in the sets for its nodes is at most $n = |S|$. Moreover, if (R_j, B_j) is a non-leaf of T, then for each of its children (R_k, B_k) we either have $r_k \leq \frac{2}{3} r_j$ and $b_k \leq b_j$, or $r_k \leq r_j$ and $b_k \leq \frac{2}{3} b_j$. This implies that the height of T is $O(\log n)$. We conclude that T can be built up in time $O(n \log n)$. Notice that both the red tree and the blue tree constructed so far have a diameter of $O(\log n)$. (This fact will be used in Section 3.) It remains to compute spanning trees for the leaves of T. Each leaf is a triangle $\triangle'$ with a monochromatic interior set X. Observe that the color of $\triangle'$ has to coincide with the color of X. The corresponding green and red (resp. green and blue) spanning trees can be computed by mainly sorting X cyclically with respect to a vertex of $\triangle'$, in time $O(n \log n)$ for all leaves. □

With the construction given so far, the spanning trees still may have a diameter of $O(n)$. We will modify the construction later on (in the proof of Theorem 5) in order to keep the diameters of the blue and the red spanning tree small.

3 General two-colored sets

We proceed to prove a generalization of Lemma 1 for point sets that have an arbitrary number of points on their convex hull.

Lemma 4 *Let $S = B \cup R$ be any two-colored point set, and let g be the number of green edges of $CH(S)$. There exists a triangulation of S that contains a green spanning tree for S, a (blue) spanning tree for B, and a (red) spanning forest for R with exactly $\max\{\frac{g}{2}, 1\}$ components.*

Proof. Let us consider the case $g = 0$ first. That is, the vertex set of $CH(S)$ is monochromatic, say blue. (The red case is symmetric.) We select some red interior point p and connect p to each vertex of $CH(S)$ with a green edge; see Figure 3 (left). This splits $CH(S)$ into blue triangles, each

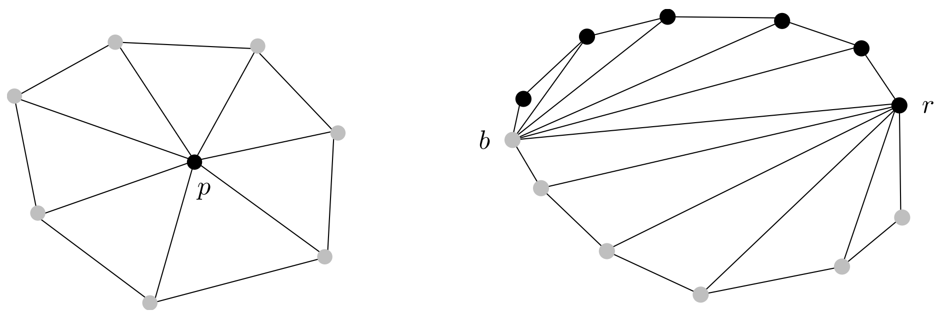

Figure 3: Fans of triangles: Blue triangles for $g = 0$ (left), blue and red triangles for $g \geq 2$ (right).

admitting a triangulation that contains red, blue, and green spanning trees as in Lemma 1. For any two such triangles, trees of the same color are already interconnected, namely blue trees via edges of $CH(S)$, and the red and the green trees all meet at point p. (We have to neglect an arbitrary blue edge of $CH(S)$ in order to avoid a blue cycle.) This proves the lemma for the case $g = 0$.

Let now $g \geq 2$. (Recall that g is an even number). We choose some blue vertex q of $CH(S)$ and connect q to other blue vertices of $CH(S)$, in a way such that $CH(S)$ is partitioned into polygons each having exactly two green edges. The number of polygons obtained from this fan of blue edges is $\frac{g}{2}$. (If $g = 2$ then the fan is empty and $CH(S)$ is the only polygon.) We triangulate each such polygon Q (if Q is not a triangle already) by introducing two fans of green edges, centered at a certain blue vertex b and a certain red vertex r of $CH(S)$, respectively, as is shown in Figure 3 (right). (Note that only one green fan will emerge if r and b are adjacent on the boundary of Q.) Each obtained triangle is either blue or red, and therefore admits a triangulation that contains red, blue, and green trees as in Lemma 1. Inside a fixed polygon Q, any two blue spanning trees (for triangles of Q) are connected via b or via blue edges of Q, any two red spanning trees are connected via r or via red edges of Q, and any two green spanning trees either have b or r in common or they are connected by the edge br. Moreover, for distinct polygons Q and Q', their blue and their green trees (but not their red trees) are connected via the blue vertex q of $CH(S)$ we have chosen above. This proves the lemma for the case $g \geq 2$. □

We are now ready to state and prove the main result of this paper.

Theorem 5 *Let $S = B \cup R$ be a two-colored set of n points. Let h be the number of vertices of $CH(S)$. In $O(n \log n)$ time, we can compute a blue crossing-free spanning tree $T(B)$ for B, and a red crossing-free spanning tree $T(R)$ for R, such that $T(B)$ and $T(R)$ cross minimally often and have diameters of $O(h + \log n)$ and $O(h \log \frac{n}{h})$, respectively.*

Proof. As a first step, let us slightly modify the construction in the proof of Lemma 1, in order to get compatible blue and red spanning trees of small diameter in the triangle case. (We do not care about the green tree now.) As in Lemma 1, assume that $\triangle$ is a blue triangle. (The case of $\triangle$ being a red triangle is symmetric.) Now, in the case $r = 0$ and $b > 0$, we simply connect each blue interior point of $\triangle$ to any of the two *blue* vertices of $\triangle$ (rather than to the red vertex). Then, by following the strategy in the proof of Lemma 3, red and blue trees of diameters $O(\log n)$ are obtained for the triangle case, in time $O(n \log n)$.

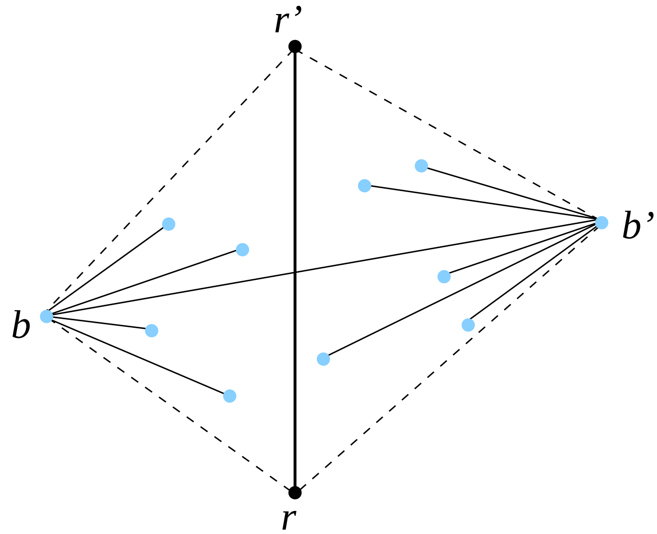

Figure 4: The red tree and the blue tree cross only once.

Based on this modification, the construction in the proof of Lemma 4 is applied. We obtain a blue crossing-free spanning tree $T(B)$ for B, and a red crossing-free spanning forest $F(R)$ for R, such that $T(B)$ and $F(R)$ are compatible. It is easy to check that the tree $T(B)$, as well as each component of $F(R)$, is of diameter $O(h + \log n)$. Let g be the number of green edges of $CH(S)$. If $g \leq 2$ then $F(R)$ is already a tree, and we are done.

Assume $g \geq 4$. Each component C of $F(R)$ lives in a subpolygon, $Q(C)$, of $CH(S)$. Adjacent subpolygons $Q(C_1)$ and $Q(C_2)$ have a blue diagonal, bb', of $CH(S)$ in common. By the construction in the proof of Lemma 1, bb' is the common edge of two blue triangles $bb'r$ and $bb'r'$, and moreover the point sets B_1 and B_2 these triangles contain have to be blue as well. See Figure 4. Observe that $bb'r \cup bb'r'$ is a convex quadrilateral, and therefore contains the line segment rr'. The tree $T(B)$ connects the points in B_1 and B_2 with blue edges to either b or b' (due to the modification above). These blue edges can be rearranged such that none of them crosses the line segment rr'. So, when we connect the red components C_1 and C_2 with the edge rr', the resulting red tree crosses $T(B)$ at a single edge, bb'. These actions for $Q(C_1)$ and $Q(C_2)$ can be carried out in $O(|B_1| + |B_2|)$ time.

In total, there are exactly $\frac{g}{2}$ consecutively adjacent subpolygons $Q(C)$. This implies that $F(R)$ can be extended to a tree $T(R)$ that crosses $T(B)$ exactly $\frac{g}{2} - 1$ times, the minimum possible by a result in [16]. The diameter of $T(R)$ is

$$O\left(g + \sum_{i=1}^{g/2}(h_i + \log n_i)\right) \text{ where } \sum_{i=1}^{g/2} h_i = O(h) \text{ and } \sum_{i=1}^{g/2} n_i = O(n).$$

This gives the bound $O(g + h + g \log \frac{n}{g})$ which, by $g < h$, is $O(h \log \frac{n}{h})$. □

Examples show that when crossings are kept to a minimum then a diameter of $\Omega(h)$ may not be avoidable. See Figure 5. The blue points and the red points are placed regularly on two concentric circles, the circle for the red points being smaller. Any blue edge which does not connect two cyclically neighbored (blue) points splits the red point set, and therefore leads to crossings with

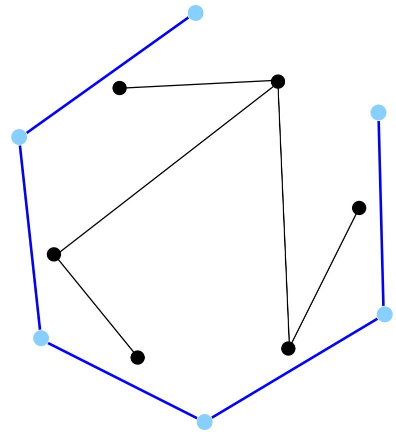

Figure 5: Lower bound for the diameter.

any possible red tree. To achieve the minimum number of crossings (which is zero in this case) the blue tree has to be a path of length $h-1$ on the boundary of the convex hull.

Let us remark another implication of Lemma 4.

Corollary 6 *Let S be a red-blue colored set of n points. Deciding whether S admits compatible blue, red, and green crossing-free spanning trees, and constructing these trees in case of their existence, can be done in $O(n \log n)$ time.*

Notice that the decision problem can be solved by simply computing $CH(S)$ and checking if the number of its green edges is at most two. Still, the $O(n \log n)$ bound is asymptotically optimal, as the vertices of $CH(S)$ have to be identified in order to make the decision, and the latter problem has a lower bound of $\Omega(n \log n)$; see [6]. Concerning the space requirement, all the described algorithms use $O(n)$ space.

4 Variants

4.1 Pointedness

A geometric straight line graph is called *pointed* if, for each of its vertices v, there is some halfplane bounded by a line through v that contains all the edges incident to v. By a result in [3], not every triangulation contains a pointed spanning tree. Let us show that both the blue spanning tree $T(B)$ and the red spanning forest $F(R)$ in Lemma 4 can be forced to be pointed while still living in a common triangulation.

Vertices of $CH(S)$ are always pointed. Thus the critical parts in the construction of $T(B)$ and $F(R)$ are those where interior points are handled. Firstly, in the case $g=0$, we may have to select some red interior point p to split $CH(S)$ into blue triangles. When choosing p at minimum distance to the boundary of $CH(S)$, p is guaranteed to stay pointed when red edges are added to it later on: All red points lie in the halfplane bounded by a line through p and being parallel to an edge of $CH(S)$. Secondly, in the construction in the proof of Lemma 1, we have to choose some red point r inside a blue triangle $\triangle$. Again, we may take r at minimum distance to the (unique) blue edge

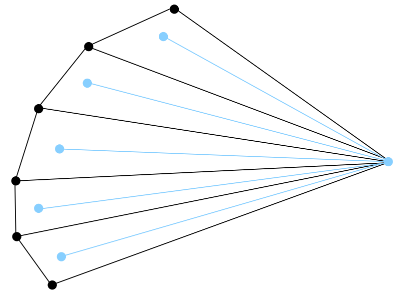

Figure 6: Many long blue edges

of △. As only red edges in the interior of △ are added to r, the pointedness of r is ensured. Finally, we have to connect each monochromatic point set within a triangle of the same color by a path. But those points have degree at most two and therefore are trivially pointed. This guarantees the pointedness of $F(R)$ with the construction just described. The arguments for the pointedness of $T(B)$ are analogous.

Observe that the corresponding green spanning tree still may be non-pointed. Note also that the computation time for $T(B)$ and $F(R)$ increases to $\Theta(n^2)$ in the worst case, as splitting no longer takes place in a balanced way.

4.2 Rigidity

A geometric graph G is called *rigid* if, loosely speaking, the only way of continuously moving any subset of its vertices without changing any edge length is to move G while keeping all vertex distances; see e.g. [8] for a rigorous definition of rigidity. G is called *minimally rigid* if G is rigid but looses this property as soon as any of its edges is deleted.

Interestingly, the blue, the red, and the green graph in Lemma 4 together yield a minimally rigid graph, M, if the red spanning forest $F(R)$ is connected to a tree in an arbitrary way. The number of edges of M is exactly $2n-3$, namely, $n-1$ in the green tree, $|B|-1$ in the blue tree, and $|R|-1$ in the red tree. Moreover, each subgraph of M with $k \geq 2$ vertices has at most $2k-3$ edges, because such a subgraph consists of a forest of each color on the respective number of vertices. That is, M is a so-called Laman graph [13, 9] and thus is minimally rigid. In fact, the red, the blue, and the green tree constitute a so-called 3T2 decomposition of M; see [10].

4.3 Extremal trees

How much length is lost, compared to the minimum spanning trees MST(R) and MST(B) of R and B, respectively, when the spanning trees $T(R)$ and $T(B)$ as in Theorem 5 are used? Figure 6 shows that the length ratio between $T(B)$ and MST(B) can be arbitrarily bad, namely, $\Theta(n)$. As a dual question, how many times do MST(R) and MST(B) cross in the worst case? Interestingly, the answer is $O(n)$, by a recent result in [11]. This shows that MST(R) and MST(B) are not too far from the optimum as far as crossings are concerned.

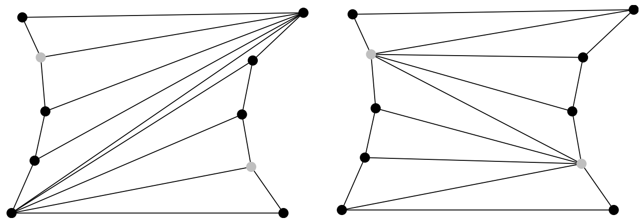

Figure 7: $\Omega(n^2)$ flips are needed to insert the blue edge

4.4 Flipping distance

The problem treated in this paper arose to us when asking for the flipping distance from the Delaunay triangulation [7] of $S = B \cup R$ to some triangulation that contains spanning trees for both subsets R and B. Clearly, this question is only meaningful if such a triangulation does exist, that is, if the red tree and the blue tree can be chosen in a compatible way.

Assume that such a triangulation does exist. We argue that $\Omega(n^2)$ flips may be necessary. In the well known example for an $\Omega(n^2)$ flipping distance between triangulations, see [7], we color two particular points blue and all other points red; see Figure 7. Then it takes $\Omega(n^2)$ flips to insert the edge connecting the two blue points. We raise the question of computing, for a given set $S = B \cup R$, a shortest flipping sequence that yields a triangulation with the properties above.

Acknowledgements Thanks go to Mark van Krefeld for mentioning to us the problem of generating colored spanning trees by flipping, and to Ferran Hurtado for pointing us to reference [16]. We also wish to thank the anonymous referees for their valuable comments.

References

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

[2] O. Aichholzer, F. Aurenhammer, P. Braß, H. Krasser. *Pseudotriangulations from surfaces and a novel type of edge flip.* SIAM Journal on Computing 32 (2003), 1621-1653.

[3] O. Aichholzer, C. Huemer, H. Krasser. *Triangulations without pointed spanning trees.* Proc. European Workshop on Computational Geometry, 2004, 221-224.

[4] E. M. Arkin, M. Held, J. S. B. Mitchell, S. S. Skiena. *Hamiltonian triangulations for fast rendering.* Visual Comput. 12 (1996), 429-444.

[5] D. Avis, H. ElGindy. *Triangulating point sets in space.* Discrete & Computational Geometry 2 (1987), 99-111.

[6] P. van Emde Boas. *On the $\Omega(n \log n)$ lower bound for convex hull and maximal vector determination.* Information Processing Letters 10 (1980), 132-136.

[7] S. Fortune. *Voronoi diagrams and Delaunay triangulations.* In: D.-Z.Du, F.Hwang (eds), Computing in Euclidean Geometry, Lecture Notes Series on Computing 4, World Scientific, 1995, 225-265.

[8] J. Graver, B. Servatius, H. Servatius. Combinatorial Rigidity. Graduate Studies in Mathematics 2, Amer. Math. Soc. 1993.

[9] R. Haas, D. Orden, G. Rote, F. Santos, B. Servatius, H. Servatius, D. Souvaine, I. Streinu, W. Whiteley. *Planar minimally rigid graphs and pseudo-triangulations.* Proc. 19th Ann. ACM Symp. Computational Geometry, 2003, 154-163.

[10] R. Haas. *Characterizations of arboricity of graphs.* Ars Combinatorica 63 (2002), 129-137.

[11] M. Kano, C. Merino, J. Urrutia. *On spanning trees and cycles of multicolored point sets with few intersections.* Information Processing Letters 93 (2005), 301-306.

[12] G. Karólyi, J. Pach, G. Tóth. *Ramsey-type results for geometric graphs, I.* Discrete & Computational Geometry 18 (1997), 247-255.

[13] G. Laman. *On graphs and rigidity of plane skeletal structures.* J. Engineering Mathematics 4 (1970), 331-340.

[14] L. C. Larson. Problem Solving Through Problems. Springer, New York, 1983.

[15] M. Overmars, J. van Leeuwen. *Maintenance of configurations in the plane.* J. Comput. System Sci. 23 (1981), 166-204.

[16] S. Tokunaga. *Intersection number of two connected geometric graphs.* Information Processing Letters 59 (1996), 331-333.