The Very Best of Perfect Non-crossing Matchings*†

Ioannis Mantas[1], Marko Savić[2], and Hendrik Schrezenmaier[3]

1 **Faculty of Informatics, Università della Svizzera italiana, Lugano, Switzerland**
`ioannis.mantas@usi.ch`

2 **Faculty of Informatics, Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Serbia**
`marko.savic@dmi.uns.ac.rs`

3 **Institut für Mathematik, Technische Universität Berlin, Germany**
`schrezen@math.tu-berlin.de`

Abstract

Given a set of points in the plane, we are interested in matching them with straight line segments. We focus on perfect (all points are matched) non-crossing (no two edges intersect) matchings. Apart from the well known MinMax variation, where the length of the longest edge is minimized, we extend work by looking into three new optimization variants such as MaxMin, MinMin, and MaxMax. We consider both the monochromatic and bichromatic versions of these problems and provide efficient algorithms for various input point configurations.

1 Introduction

In the *matching problem*, we are given a set of objects and the goal is to partition the set into pairs such that no object belongs to two pairs. This simple problem is a classic in graph theory, which has received a lot of attention, both in an abstract and in a geometric setting.

In this paper, we consider the geometric setting where given a set P of $2n$ points in the plane, the goal is to match points of P with straight line segments, in the sense that each pair of points induces an *edge* of the matching. A matching is *perfect* if it consists of exactly n pairs. A matching is *non-crossing* if the edges of the matching are pairwise disjoint. When there are no restrictions on which points can be matched, the problem is called *monochromatic*. In the *bichromatic* variant, P is partitioned into two sets B and R of blue and red points, respectively, and only points of different colors are allowed to be matched. When $|B| = |R| = n$, the point set P is called *balanced*.

Related work on perfect non-crossing matchings. Geometric matchings find applications in various areas, as in operations research, in the field of shape matching, in pattern recognition, in VLSI design problems and map construction or comparison algorithms among others. In any application, requiring the matching to be non-crossing or perfect, seems natural. Given a point set, monochromatic or balanced bichromatic, a perfect non-crossing matching always exists and it can be found in $O(n \log n)$ time by recursively computing *ham-sandwich cuts* [13] or by using the algorithm of Hershberger and Suri [12].

* I. M. was partially supported by the Swiss National Science Foundation, project SNF 200021E-154387, M. S. was partially supported by Ministry of Education, Science and Technological Development, Republic of Serbia, project 174019, and H. S. was partially supported by the German Research Foundation, DFG grant FE-340/11-1.

† A full version of this extended abstract is available at `https://arxiv.org/abs/2001.03252`.

36th European Workshop on Computational Geometry, Würzburg, Germany, March 16–18, 2020.

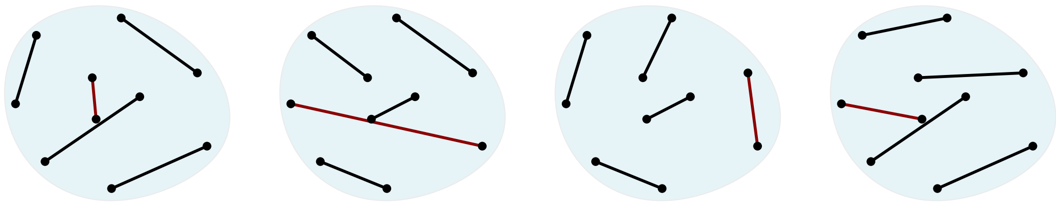

Figure 1 Optimal MinMin1, MaxMax1, MinMax1, and MaxMin1 matching of a point set.

A well-studied optimization criterion is minimizing the sum of lengths of all edges, referred to as MinSum, and also known as the *Euclidean assignment* or *matching* problem. It is interesting, and easy to see, that such a matching is always non-crossing. For monochromatic point sets, an $O(n^{3/2} \log n)$-time algorithm was given by Varadarajan [19]. For bichromatic point sets, Agarwal et al. [2] presented an $O(n^{2+\varepsilon})$-time algorithm. When points are in convex position, Marcotte and Suri [14] solved the problem in $O(n \log n)$ time in both settings.

Another popular goal is to minimize the length of the longest edge, which we refer to as the MinMax variant and is also known as the *bottleneck matching*. For monochromatic point sets, Abu-Affash et al. [1] showed that finding such a matching is $\mathcal{NP}$-hard and accompanied this with an $O(n^3)$-time algorithm for points in convex position. This was recently improved to $O(n^2)$ time by Savić and Stojaković [15]. For bichromatic point sets, Carlsson et al. [7] showed that finding such a matching is also $\mathcal{NP}$-hard. Biniaz et al. [6] gave an $O(n^3)$-time algorithm for points in convex position and an $O(n \log n)$-algorithm for points on a circle. These were recently improved to $O(n^2)$ and $O(n)$, respectively, by Savić and Stojaković [16].

More optimization goals have been studied, as the *uniform* or *fair matching*, where the goal is to minimize the length difference between the longest and the shortest edge, and the *minimum deviation matching*, where the difference between the shortest edge length and the average edge length should be minimized. Both can be solved in polynomial time, see Efrat et al. [10, 11]. Another example is the MaxSum variant, where the goal is to maximize the sum of the edge lengths. Alon et al. [4] conjectured that the problem is $\mathcal{NP}$-hard.

Problem variants considered and our contribution. We continue exploring similar optimization variants in different settings. We consider solely perfect non-crossing matchings, without further mention. We deal with four variants: MinMin where the length of the shortest edge is minimized, MaxMax where the length of the longest edge is maximized, MaxMin where the length of the shortest edge is maximized, and MinMax, see Fig. 1.

To the best of our knowledge, out of the four variants, only MinMax, has been considered before. Apart from the theoretical interest, MinMin and MaxMax are motivated by worst-case analyses of matchings, where there is no control over the choice of edges, and short or long edges are undesirable. Apart from the applications of MinMax, together with MaxMin, such matchings resemble fair matchings in the sense that all edges have similar lengths.

We investigate both the monochromatic and bichromatic versions of these variants. In the bichromatic version, we assume that P is balanced. We denote the monochromatic problems with the index 1, e.g., MinMin1, and the bichromatic with the index 2, e.g., MinMin2.

These problems are examined in different point configurations. In Section 2, we consider points in general position. In Section 3, points are in convex position. In Section 4, points lie on a circle. In Section 5, we consider *doubly collinear* bichromatic point sets, where the blue points lie on one line and the red points on another line. By studying structural properties of each variant and combining diverse techniques, we come up with a series of interesting results that are summarized in Table 1.

Table 1 The running times for finding the values of optimal matchings regarding different objective functions. The times marked with (*) indicate the extra time needed to return also the matching. h denotes the size of the convex hull. Results without reference are given in this paper.

Monochromatic	**MinMin1**	**MaxMax1**	**MinMax1**	**MaxMin1**
General Position	$O(nh + n\log n)$	$O(nh) + O(n\log n)^*$	$\mathcal{NP}$-hard [1]	?
Convex Position	$O(n)$	$O(n)$	$O(n^2)$ [15]	$O(n^3)$
Points on circle	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Bichromatic	**MinMin2**	**MaxMax2**	**MinMax2**	**MaxMin2**
General Position	?	?	$\mathcal{NP}$-hard [7]	?
Convex Position	$O(n)$	$O(n)$	$O(n^2)$ [16]	$O(n^3)$
Points on circle	$O(n)$	$O(n)$	$O(n)$ [16]	$O(n^3)$
Doubly collinear	$O(n)$	$O(1) + O(n)^*$	$O(n^4 \log n)$	?

2 Monochromatic points in general position

In this section, P is a set of points in general position, where we assume that no three points are collinear. We denote by $\mathrm{CH}(P) \subseteq P$ the set of points on the boundary of the convex hull of P and set $h := |\mathrm{CH}(P)|$. By $d(v, w)$ we denote the Euclidean distance of two points v, w.

An edge (v, w) of points is *feasible* if there exists a matching which contains (v, w).

▶ **Lemma 2.1.** *An edge (v, w) is infeasible if and only if $v, w \in \mathrm{CH}(P)$ and there is an odd number of points on each side of the line through (v, w). See Fig. 2a.*

This criterion can be checked efficiently using the *radial orderings* of the points $p \in P$, i.e., the circular orderings of the points in $P \setminus p$ by angle around p. The radial orderings of all points in P can be computed in $O(n^2)$ time, see, e.g., [3, 5]. Instead, we define the *weak radial orderings*. Given a set $A \subseteq P$, in the *A-weak radial ordering* of p the points from A occurring between two points from $\overline{A} := P \setminus A$ are given as an unordered set, see Fig. 2b-c.

▶ **Lemma 2.2.** *The $\overline{A}$-weak radial orderings of all points in A can be computed in $O(n|A|)$ time.*

MinMin1. We are looking for the shortest feasible edge. We first compute $\mathrm{CH}(P)$ in $O(n \log h)$ time [8] and the $\overline{\mathrm{CH}(P)}$-weak radial orderings of points in $\mathrm{CH}(P)$ in $O(nh)$ time. Then, we find $m_1 := \min(\{ d(v, w) \colon v \in P \setminus \mathrm{CH}(P), w \in P \})$ in $O(n \log n)$ time using a Voronoi diagram and $m_2 := \min(\{ d(v, w) \colon v, w \in \mathrm{CH}(P) \})$ in $O(nh)$ time using weak radial orderings. The solution is then $m_{\mathrm{sol}} = \min(m_1, m_2)$, resulting in the following theorem.

▶ **Theorem 2.3.** MinMin1 *can be solved in $O(nh + n \log n)$ time.*

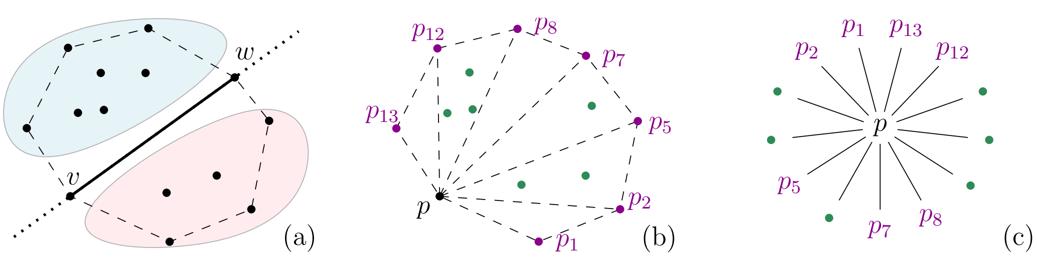

Figure 2 (a) An infeasible edge (v, w). (b-c) The $\overline{\mathrm{CH}(P)}$-weak radial ordering of a point p.

MaxMax1. We can use the same $O(nh+n\log n)$-time algorithm, considering maximizations in m_1, m_2, and m_{sol} instead of minimizations. Using Lemma 2.4 we can reduce the time needed to find m_1 in $O(nh)$, by simply comparing all $(n-h)h$ edges. This results in Theorem 2.5.

▶ **Lemma 2.4.** *If (v, w) is a longest feasible edge, then one of $v, w \in \mathrm{CH}(P)$.*

▶ **Theorem 2.5.** MaxMax1 *can be solved in $O(nh)$ time.*

3 Points in convex position

In this section, we assume that points in P are in convex position with the counterclockwise ordering, $p_0, \ldots, p_{2n-1}$, given. We address a point p_i by its index i, and do arithmetic operations modulo $2n$. We call edges of the form $(i, i+1)$ *boundary edges*.

Dynamic programming. We can easily solve all four optimization variants in $O(n^3)$ time using a classic dynamic programming approach, which is also used for MinMax [1, 6, 7].

▶ **Theorem 3.1.** *If P is convex,* MaxMin1 *and* MaxMin2 *can be solved in $O(n^3)$ time.*

3.1 MinMin and MaxMax matchings in convex position

Monochromatic. We split P into two (convex) sets, P_{odd} and P_{even}, according to the parity of the indices. A pair (i, j) is feasible if and only if i and j are of different parity [1]. Hence, any (i, j) with $i \in P_{\mathrm{odd}}$ and $j \in P_{\mathrm{even}}$ is feasible. Given two convex sets P, Q, we can find in $O(|P| + |Q|)$ time the points that realize the minimum [18] and the maximum [9] distance between them. Applying these algorithms to P_{odd} and P_{even}, we obtain the following.

▶ **Theorem 3.2.** *If P is convex,* MinMin1 *and* MaxMax1 *can be solved in $O(n)$ time.*

Bichromatic. We now combine the monochromatic algorithms with the theory of *orbits* [16], a concept which captures well the nature of bichromatic matchings in convex position. More specifically, P can be partitioned in $O(n)$ time into orbits, which are balanced sets of points. In each orbit point colors alternate and a bichromatic edge (b, r) is feasible if and only if b and r are in the same orbit. Thus, to each orbit separately we can use the algorithms of [9, 18] and return the overall longest or shortest edge, see Fig. 3, resulting in the following.

▶ **Theorem 3.3.** *If P is convex,* MinMin2 *and* MaxMax2 *can be solved in $O(n)$ time.*

Given an extremal feasible edge, we can extend it to an optimal matching, in $O(n)$ time, by applying the following lemma to the sets $\{i+1, \ldots, j-1\}$ and $\{j+1, \ldots, i-1\}$.

▶ **Lemma 3.4.** *If P is in convex position, we can construct an arbitrary matching in $O(n)$ time, both in the monochromatic and bichromatic case.*

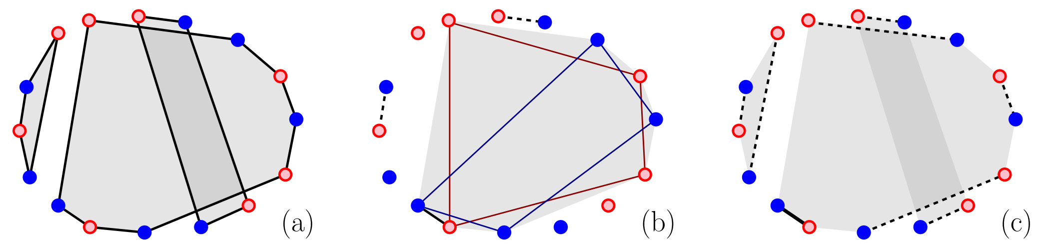

Figure 3 MinMin2 for a set in convex position. (a) Find orbits. (b) Find the shortest edge between the blue and red polygon of an orbit. (c) Extend to a perfect matching using Lemma 3.4.

4 Points on a circle

Next, we assume that all points of P lie on a circle. Obviously, the results from Section 3 also apply here. Apart from convexity, these results rely only on a property of point sets lying on a circle, which we call the *decreasing chords property*. A point set P has this property if, for any edge (i, j), for at least one of its sides, all the possible edges between two points on that side are not longer than (i, j) itself, see Fig. 4a. Using this, we can easily infer the following.

▶ **Lemma 4.1.** *Any shortest edge of a matching on P is a boundary edge.*

MinMin1 and MinMin2. Due to Lemma 4.1, these can be solved in $O(n)$ time significantly simpler, without using Theorems 3.2 and 3.3, by finding the shortest feasible boundary edge.

MinMax1. We show that there exists a MinMax1 matching using only boundary edges. There are two such matchings and we find the optimal in $O(n)$ time, resulting in the following.

▶ **Theorem 4.2.** *If P lies on a circle,* MinMax1 *can be solved in $O(n)$ time.*

MaxMin1. Lemma 4.1 suggests an approach for MaxMin1 by *forbidding* short boundary edges and checking if we can find a matching without them. Let some boundary edges be *forbidden* and the remaining be *allowed*. A *forbidden chain* is a maximal sequence of consecutive forbidden edges. See Fig. 4b for an example (with forbidden edges shown dashed).

▶ **Lemma 4.3.** *There exists a matching without the forbidden edges if and only if $l < n$, where l is the length of a longest forbidden chain.*

MaxMin1 is equivalent to finding the largest value μ such that there exists a matching with all edges of length at least μ. By Lemma 4.1, it suffices to search for μ among the lengths of the boundary edges. By Lemma 4.3, this means that we need to find the maximal length μ of a boundary edge such that there are no n consecutive boundary edges all shorter than μ. We can find μ as follows. Consider all $2n$ sets of n consecutive boundary edges and associate to each set the longest edge in it. Then, out of the $2n$ longest edges, we search for the shortest one. This fits under the *sliding window maximum problem*, for which several simple algorithms are known, see, e.g., [17]. Adapting to our problem we obtain the following.

▶ **Theorem 4.4.** *If P lies on a circle,* MaxMin1 *can be solved in $O(n)$ time.*

Using Lemma 4.5, we can construct an optimal matching within the same time complexity.

▶ **Lemma 4.5.** *Given a value $\mu > 0$, a matching consisting of edges of length at least μ can be constructed in $O(n)$ time if it exists.*

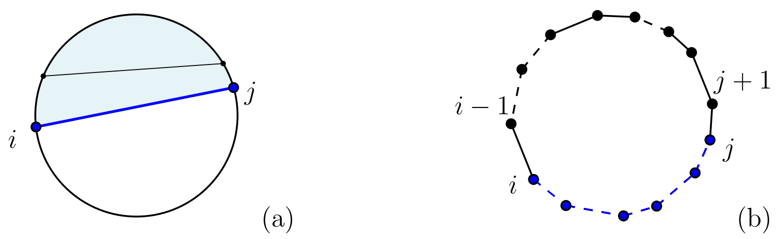

Figure 4 (a) The decreasing chords property. (b) Example of a forbidden chain $\{i, \ldots, j\}$.

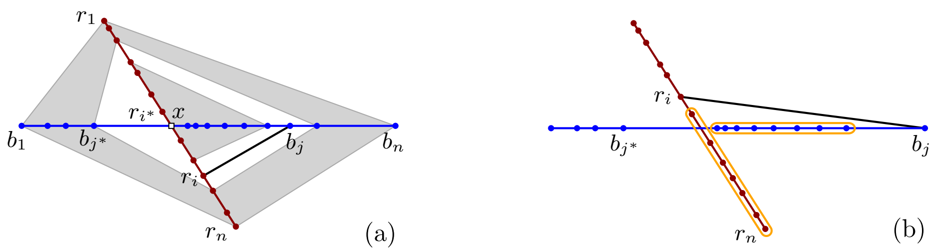

Figure 5 A doubly collinear point set, with (a) a feasible and (b) an infeasible edge shown.

5 Doubly collinear points

A bichromatic point set P is *doubly collinear* if the blue points lie on a line l_B and the red points lie on a line l_R. We assume that l_B and l_R are not parallel and that the ordering of the points along each line is given. Let $x = l_B \cap l_R$ and assume, for simplicity, that $x \notin P$.

5.1 MinMin2 and MaxMax2 matchings on doubly collinear points

We first give a feasibility criterion for an edge, see Fig. 5, which can be checked in $O(1)$ time. It also indicates an $O(n)$-time algorithm, to construct a matching, given a feasible edge (r, b).

▶ **Lemma 5.1.** *Let $r_1, \ldots, r_i, \ldots, r_{i^*}, x, r_{i^*+1}, \ldots, r_n$ and $b_1, \ldots, b_j, \ldots, b_{j^*}, x, b_{j^*+1}, \ldots, b_n$ be the points on l_R and l_B, respectively, in sorted order. Then, the edge (r_i, b_j) is feasible if and only if $i^* - i \le n - j$ and $j^* - j \le n - i$.*

As a consequence of Lemma 5.1, the order of the closest feasible neighbors of the red points on the line l_B coincides with the order of the red points on l_R. Thus, we can find the closest feasible neighbor of all red points in total $O(n)$ time, implying the following theorem.

▶ **Theorem 5.2.** *If P is doubly collinear,* MinMin2 *can be solved in $O(n)$ time.*

The same algorithm solves MaxMax2. We show that the longest feasible edge is an edge between the, at most four, points on CH(P), so we can further improve as follows.

▶ **Theorem 5.3.** *If P is doubly collinear,* MaxMax2 *can be solved in $O(1)$ time.*

5.2 MinMax2 and MaxMin2 matchings on doubly collinear points

One-sided case. We first consider the case, where all red points are on one side of l_B. This can be solved via a dynamic program with $O(n^2)$ subproblems in total, see Fig. 6.

▶ **Theorem 5.4.** *If P is one-sided doubly collinear,* MaxMin2 *can be solved in $O(n^2)$ time.*

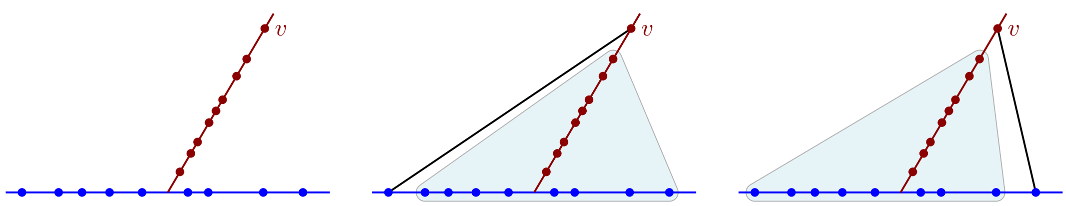

Figure 6 The one-sided case with the two possibilities to match v and the resulting subproblems.

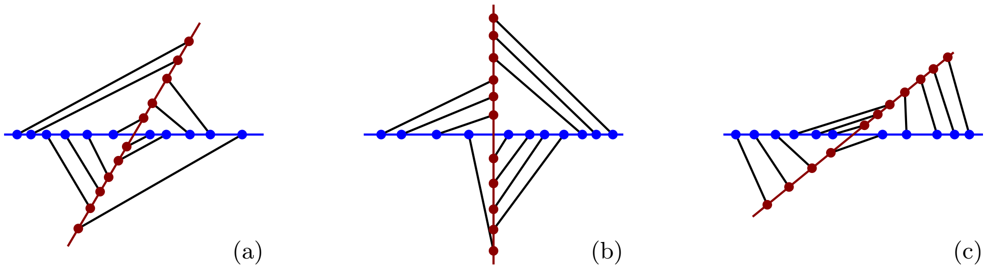

Figure 7 Optimal matchings having a special structure (a) for MinMax2, (b) for MinMax2 and MaxMin2 if $\alpha = \frac{\pi}{2}$, and (c) for MinMax2 if $\alpha < \frac{\pi}{4}$.

In the case of MinMax2 we can further improve upon this, by proving (also for the two-sided case) the existence of an optimal matching with a special form where the points are partitioned into a constant number of blocks and these blocks are matched, see Fig. 7a.

▶ **Theorem 5.5.** *If P is one-sided doubly collinear,* MinMax2 *can be solved in $O(n \log n)$ time.*

General case. For MinMax2, the aforementioned form can also be applied to the general case, yielding Theorem 5.6. On the contrary, for MaxMin2 we are not aware if an optimal matching with a special form exists. Thus, we do not know a polynomial time algorithm.

▶ **Theorem 5.6.** *If P is doubly collinear,* MinMax2 *can be solved in $O(n^4 \log n)$ time.*

Special intersection angle. Let α be the angle of intersection of l_B and l_R. By proving the existence of optimal matchings of a special form, see Fig. 7b-c, we can obtain the following.

▶ **Theorem 5.7.** *If $\alpha = \frac{\pi}{2}$,* MinMax2 *and* MaxMin2 *can be solved in $O(n)$ time.*

▶ **Theorem 5.8.** *If $\alpha \leq \frac{\pi}{4}$,* MinMax2 *can be solved in $O(n)$ time.*

6 Concluding remarks

It comes as no surprise that the MaxMin variant exhibits this significant difficulty; devising efficient algorithms even for simple configurations is not at all obvious and, hence, interesting on its own. On the contrary, the MinMin and the MaxMax variants are *relatively easier* to tackle; we managed to design optimal algorithms by exploiting structural properties combined with existing techniques from diverse problems. We conclude with some open questions, hoping to see Table 1 filled in. For bichromatic P in general position, can we check the feasibility of an edge in polynomial time? This would imply polynomial time algorithms for MinMin2 and MaxMax2. For P in convex position, do there exist $o(n^3)$-time algorithms for MaxMin? Maybe by using the theory of orbits for the bichromatic case? For P in general position, can MaxMin be solved in polynomial time? What if P is doubly collinear?

Acknowledgements. Preliminary results were obtained during the IRP - DCCG (Barcelona, 4-6/2018). We are grateful to CRM, UAB for hosting the event and to the organizers for providing us with the platform to meet and collaborate. We are also grateful to Carlos Alegría, Carlos Hidalgo Toscano, Oscar Iglesias Valiño, and Leonardo Martínez Sandoval for initial discussions, and to Carlos Seara for raising a question that motivated this work.

References

1. A. Karim Abu-Affash, Paz Carmi, Matthew J. Katz, and Yohai Trabelsi. Bottleneck non-crossing matching in the plane. *Computational Geometry*, 47(3A):447–457, 2014.
2. Pankaj K. Agarwal, Alon Efrat, and Micha Sharir. Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. *SIAM Journal on Computing*, 29(3):912–953, 2000.
3. Pankaj K. Agarwal and Micha Sharir. Arrangements and their applications. In *Handbook of Computational Geometry*, chapter 2, pages 49–119. North-Holland, 2000.
4. Noga Alon, Sridhar Rajagopalan, and Subhash Suri. Long non-crossing configurations in the plane. In *Proc. 9th Annual Symposium on Computational Geometry*, pages 257–263, 1993.
5. Tetsuo Asano, Subir K. Ghosh, and Thomas C. Shermer. Visibility in the plane. In *Handbook of Computational Geometry*, chapter 19, pages 829–876. North-Holland, 2000.
6. Ahmad Biniaz, Anil Maheshwari, and Michiel H.M. Smid. Bottleneck bichromatic plane matching of points. In *Proc. 26th Canadian Conference on Computational Geometry*, pages 431–435, 2014.
7. John G. Carlsson, Benjamin Armbruster, Saladi Rahul, and Haritha Bellam. A bottleneck matching problem with edge-crossing constraints. *International Journal of Computational Geometry & Applications*, 25(4):245–261, 2015.
8. Timothy M. Chan. Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete & Computational Geometry*, 16(4):361–368, 1996.
9. Herbert Edelsbrunner. Computing the extreme distances between two convex polygons. *Journal of Algorithms*, 6(2):213–224, 1985.
10. Alon Efrat, Alon Itai, and Matthew J. Katz. Geometry helps in bottleneck matching and related problems. *Algorithmica*, 31(1):1–28, 2001.
11. Alon Efrat and Matthew J. Katz. Computing fair and bottleneck matchings in geometric graphs. In *Proc. 7th International Symposium on Algorithms and Computation*, pages 115–125. Springer, 1996.
12. John Hershberger and Subhash Suri. Applications of a semi-dynamic convex hull algorithm. *BIT Numerical Mathematics*, 32(2):249–267, 1992.
13. Chi-Yuan Lo, Jiří Matoušek, and William Steiger. Algorithms for ham-sandwich cuts. *Discrete & Computational Geometry*, 11(4):433–452, 1994.
14. Odile Marcotte and Subhash Suri. Fast matching algorithms for points on a polygon. *SIAM Journal on Computing*, 20(3):405–422, 1991.
15. Marko Savić and Miloš Stojaković. Faster bottleneck non-crossing matchings of points in convex position. *Computational Geometry*, 65:27–34, 2017.
16. Marko Savić and Miloš Stojaković. Bottleneck bichromatic non-crossing matchings using orbits, 2018. URL: `arxiv.org/abs/1802.06301`, `arXiv:1802.06301`.
17. Kanat Tangwongsan, Martin Hirzel, and Scott Schneider. Low-latency sliding-window aggregation in worst-case constant time. In *Proc. 11th ACM International Conference on Distributed and Event-based Systems*, pages 66–77, 2017.
18. Godfried T. Toussaint. An optimal algorithm for computing the minimum vertex distance between two crossing convex polygons. *Computing*, 32(4):357–364, 1984.
19. Kasturi R. Varadarajan. A divide-and-conquer algorithm for min-cost perfect matching in the plane. In *Proc. 39th Symposium on Foundations of Computer Science*, pages 320–329, 1998.