A bottleneck matching problem with edge-crossing constraints

John Gunnar Carlsson and Benjamin Armbruster

October 23, 2010

Abstract

Motivated by a crane assignment problem, we consider a Euclidean bipartite matching problem with edge-crossing constraints. Specifically, given \( n \) red points and \( n \) blue points in the plane, we want to construct a perfect matching between red and blue points that minimizes the length of the longest edge, while imposing a constraint that no two edges may cross each other. We show that the problem is NP-hard and give an algorithm that solves our problem in \( O(n^4 \log n) \) time when the red and blue points form the vertices of a convex polygon.

1 Introduction

Our problem can be motivated by a problem in port operations. Consider an equal number of cranes and containers in the plane. Each crane \( i \) must be assigned to a container \( j \). However, crane \( i \) can only be assigned to containers within a distance \( R \) from it and no two cranes may cross each other. This as a Euclidean bipartite matching problem: given a collection of \( n \) red points \( \{r_1, \ldots, r_n\} \) (the cranes) and a collection of \( n \) blue points \( \{b_1, \ldots, b_n\} \) (the containers), our goal is to find a crossing-free matching between red-blue pairs. We call such a matching a Euclidean non-crossing bipartite matching (ENCBM) and denote the size of the matching as the length of the longest edge, (i.e., the bottleneck size). Thus our decision problem is to determine whether an ENCBM of size at most \( R \) exists, and our optimization problem is to find an ENCBM of minimal size. This optimization problem always has a feasible solution because a minimum-weight Euclidean matching (i.e., a matching that minimizes the total length of the edges) is always crossing-free [5]. In fact, it is easy to see that the minimum Euclidean matching is a factor \( n \) approximation algorithm to our problem, as shown in Figure 1.

![Figure 1: A tight example showing the factor \( n \) approximation of our problem given by the minimum-weight Euclidean matching: (b) has a maximum edge length of 1 while (c) has a maximum edge length of \( (n - 2)(1 + \epsilon) \).](image)

Previous work The problem of finding a crossing-free configuration of a graph in the plane is well-studied. The paper [3] considers the problem of determining whether a crossing-free spanning tree,
perfect matching, or two-factor exists in a planar embedding of a graph under various restrictions on the segments that connect points. Most closely related to our result is their proof that the problem of finding a (non-bipartite) crossing-free perfect matching in a set of points on the lattice $\mathbb{Z} \times \mathbb{Z}$ is NP-hard when we restrict ourselves to only edges of constant length $d \in \mathbb{Z}$. The paper [6] shows that reconstructing a set of $n$ orthogonal line segments in the plane from their set of endpoints can be done in $O(n \log n)$ time if the segments are allowed to cross, and if the segments are not allowed to cross, then the problem is NP-hard. The paper [1] shows that, if we are given $r$ red points in the plane and $b$ blue points in the plane, then there exists a crossing-free matching of points of the same color that matches at least $83.33\%$ of the points and can be found in $O((r+b) \log (r+b))$ time.

Detection of crossing-free subgraphs is a useful problem in several domains. The paper [8] considers the problem of scheduling a set of cranes to cover a set of jobs when a non-crossing constraint (among other spatial considerations) is imposed. The authors assume that the cranes and containers are located in two parallel columns. The paper [9] extends this; proves that the scheduling problem is NP-hard; and gives a branch-and-bound algorithm for solving it. Crossing-free configurations are relevant in VLSI applications where the number of crossings in a matching corresponds to the number of layers required in the layout of the circuit design [2].

### New results
This paper presents two new results about the decision problem of determining whether an ENCBM of size at most $R$ exists: first, we show that it is NP-hard, and second, we give an algorithm solving the case when the points form the vertices of a convex polygon in $O(n^4)$ time. This algorithm can also be extended to the non-bipartite case and to the case where all points are the vertices of a simple polygon and we use the geodesic distance between points.

## 2 Hardness

In this section we show that the decision problem “does there exist a non-crossing perfect matching of a given red-blue configuration with all edge lengths at most $R$?” reduces to the planar 3-satisfiability problem (planar 3-SAT), which is known to be NP-hard [4]. Paper [6] also uses a reduction to planar 3-SAT on a problem dealing with non-crossing orthogonal edges linking points and no constraints on distances. Our proof essentially uses the same reduction although we require a few new ideas to tie everything together.

### Planar 3-SAT
Let us define an instance of planar 3-SAT with variables $X := \{x_1, \ldots, x_m\}$, and clauses $C := \{c_1, \ldots, c_n\}$. We denote negation with a tilde, e.g., $\bar{x}_1$. As in normal 3-SAT, each clause is a logical disjunction of three variables or their negations, e.g., $c_1 := x_1 \lor x_5 \lor \bar{x}_7$. We now define a graph whose vertices are the variables and clauses, $X \cup C$, and whose edges connect any variable and the clauses it appears in, $E := \{(x_i, c_j) | x_i \in c_j \text{ or } \bar{x}_i \in c_j\}$. In planar 3-SAT, this graph, $(X \cup C, E)$ is planar. (The definition of planar 3-SAT in [4] has some additional edges between the vertices in $X$. That this larger graph is planar obviously implies that our smaller graph, $(X \cup C, E)$, is planar.)

### Reduction
Our reduction will produce a set of red and blue points $V$ such that an ENCBM of size at most $R$ exists if and only if our planar 3-SAT instance, $(X \cup C, E)$ has a solution. For any set of red and blue points, $P$, we define $G(P)$ as the graph formed by constructing all edges between red and blue points with length at most $R$. Thus an ENCBM of size at most $R$ exists on a set $P$ if and only if $G(P)$ has a perfect matching.
Variables Our reduction will replace any variable \( x_i \) with a large collection of red and blue points \( V_i \) arranged in a loop \( \ell_i := G(V_i) \) with the points a distance \( R \) away from each other and alternating in color as in Figure 2. (For now, we omit the discussion of exactly how many points are needed.) This configuration clearly has two possible perfect matchings: one with red points matched to blue points in the clockwise direction and one in the counterclockwise direction. We decide arbitrarily that matching red points to blue in the clockwise direction corresponds to setting variable \( x_i \) true.

Edges We treat each edge in the planar 3-SAT between a variable, \( x_i \), and a clause \( c_j \), \( (x_i, c_j) \in E \), like a variable whose value is \( x_i \) or \( \overline{x_i} \) if \( x_i \) or \( \overline{x_i} \) appears in \( c_j \), respectively. As with a variable, we thus represent an edge in the planar 3-SAT by a collection of red and blue points \( V_{ij} \) arranged in a loop \( \ell_{ij} := G(V_{ij}) \) with the points a distance \( R \) away from each other and alternating in color. As in the preceding, the orientation of the perfect matching on \( \ell_{ij} \) corresponds to the truth-value of the edge. To enforce this connection we intersect the loops \( \ell_i \) and \( \ell_{ij} \) as in Figure 3.

Loop intersection We now describe how loop \( \ell_{ij} \) will intersect the loop \( \ell_i \) in such a way that the matching on \( \ell_i \) will determine the matching on \( \ell_{ij} \). At each intersection of \( \ell_{ij} \) and \( \ell_i \), \( G(V_{ij} \cup V_i) \) has two additional edges connecting red and blue points on \( \ell_i \) to points of the opposite color on \( \ell_{ij} \) as shown in Figure 4. This is problematic because the matching on \( \ell_i \) does not affect the matching on \( \ell_{ij} \) as shown in Figure 5. Fortunately, we can resolve this issue by inserting two additional red points and two additional blue points at each intersection (see Figure 6), and therefore we can assume from now on that the orientations of all loops \( \ell_{ij} \) are imposed by the orientation of \( \ell_i \).

Clauses Let us examine clause \( c_j \) and assume without loss of generality that \( c_j \) is connected by three edges to \( x_1 \), \( x_2 \), and \( x_3 \) the planar 3-SAT. Then, with only a slight abuse of notation, \( c_j = \ell_{1j} \lor \ell_{2j} \lor \ell_{3j} \). In Figure 7 we represent each clause \( c_j \) as a rectangle \( \square_j \).

The construction of \( \square_j \) is shown in Figure 8. Each loop \( \ell_{1j}, \ell_{2j}, \ell_{3j} \) enters \( \square_j \) on the bottom side. Let \( \Sigma_j \) denote a union of three alternating red-blue loops in \( \square_j \) that cross each \( \ell_{ij} \). Finally, let \( \Pi_j \) denote an alternating red-blue loop in \( \square_j \) that intersects each of the three loops in \( \Sigma_j \). Whenever two loops intersect in this construction, we “fix” the intersection using two additional red points and two additional blue points as described above. It is now easy to verify that a non-crossing matching of the points in \( \square_j \) exists if and only if one of \( \ell_{1j}, \ell_{2j}, \) or \( \ell_{3j} \) is true, as shown in Figures 9 and 10.
Figure 3: A collection of loops representing the connections between variable $x_1$ and its associated clauses $c_1$, $c_2$, and $c_3$.

Figure 4: Each intersection of $\ell_{ij}$ and $\ell_i$, creates two additional edges in $G(V_{ij} \cup V_i)$ between the two loops.
Figure 5: Two perfect matchings of the loop $\ell_{ij}$ given the same matching on $\ell_i$.

Figure 6: By inserting two additional red points and two additional blue points we ensure that the matching on $\ell_i$ induces the desired matching on $\ell_{ij}$. This is because the two newly inserted red points have only one (blue) neighbor each and therefore must be connected to that neighbor in a perfect matching. The non-crossing constraint eliminates the dotted edges, which leaves us with the desired intersection structure.
Figure 7: An arrangement of loops corresponding to a variable $x_1$ and four clauses $c_1, c_2, c_3, c_4$ containing it. Notice that $\ell_{11}$ must have the same matching orientation as $\ell_1$ but $c_2$, $c_3$, and $c_4$ have opposing orientations to $\ell_1$. This corresponds to the case where $c_1$ uses $x_1$ and $c_2$, $c_3$, and $c_4$ use $\bar{x}_1$. 
Figure 8: The contents of rectangle $\Box_j$, which corresponds to clause $c_j = \ell_{1j} \lor \ell_{2j} \lor \ell_{3j}$. As shown in the middle diagram, setting $\ell_{1j}$, $\ell_{2j}$, and $\ell_{3j}$ false ensures that no perfect matching can exist in loop $\Pi_j$. This is because the required matching on $\Sigma_j$ prevents a perfect matching from occurring in the shaded ellipse shown. On the other hand, setting the three variables true allows a perfect matching to exist as shown in the third diagram.
Figure 9: The perfect matchings in $\square_j$ corresponding to the cases where one of $\ell_{1j}$, $\ell_{2j}$, and $\ell_{3j}$ is true.
Figure 10: The perfect matchings in $\Box_j$ corresponding to the cases where two of $\ell_{1j}$, $\ell_{2j}$, and $\ell_{3j}$ are true.
To complete the proof, we merely have to show that the the number of red and blue points that are required depends on the number of variables \( m \) and the number of clauses \( n \) in a polynomial fashion. A well-known theorem of Schnyder [7] says that any planar graph with \( k \) vertices has an embedding on the \( k - 2 \times k - 2 \) grid. We thus embed our planar 3-SAT on the \((n + m - 2) \times (n + m - 2)\) grid. We then place the loops \( \ell_i \) and \( \ell_{ij} \) and boxes, \( \square_j \), in the rough locations of the corresponding vertices and edges of the 3-SAT graph on the grid. Each \( \square_j \) contains a constant number of red and blue points and thus has a diameter \( \mathcal{O}(R) \). Since the edges in the grid have length less than \((n + m - 2)\sqrt{2} \) the loops \( \ell_{ij} \) have \( \mathcal{O}((n + m)/R) \) nodes. The loops \( \ell_i \) corresponding to variables are connected to at most \( m \) edges and thus contain \( \mathcal{O}(m) \) red and blue points and has diameter \( \mathcal{O}(m R) \). To ensure that loops \( \ell_i \) and boxes \( \square_j \) have diameter (say) 1/2, we choose \( R \) of \( \mathcal{O}(1/m) \). Thus a loop such as \( \ell_{ij} \), of which there are \( 3n \), has \( \mathcal{O}(m(n + m)) \) nodes. Therefore, the total number of red and blue points is \( \mathcal{O}((n + m)mn) \).

3 An algorithm for the convex case

In this section we consider the special case when all points \( \{r_1, \ldots, r_n\} \) and \( \{b_1, \ldots, b_n\} \) form the vertices of a convex polygon. We give an algorithm that decides whether an ENCBM of size at most \( R \) exists in running time \( \mathcal{O}(n^4) \). This allows us to find the smallest ENCBM in \( \mathcal{O}(n^4 \log n) \) time using bisection on \( R \). Let \( \{x_1, \ldots, x_{2n}\} \) denote the red and blue vertices of this polygon in counter-clockwise order with an arbitrary first vertex \( x_1 \). Connect all pairs of vertices with an edge \( (x_i, x_j) \) if \( \|x_i - x_j\| \leq R \) and \( x_i \) and \( x_j \) are different colors. We will now abuse notation and use \((i, j)\) to refer both to an interval in \( \mathbb{Z} \) and to the edge \( (x_i, x_j) \). Two edges \((i, j)\) and \((s, t)\) with \( i < j \) and \( s < t \) are non-crossing if and only if one interval is contained within the other, or if the intervals are entirely disjoint, equivalently the edges are crossing if and only if \( i \leq s \leq j \leq t \) or \( s \leq i \leq t \leq j \).

We first sort all edges \( e_k := (i_k, j_k) \) by their width, \( j_k - i_k \). For any given \( k \) and interval \((s, t)\) define \( I(k, s, t) \) as the subset of edges \( \{e_1, \ldots, e_k\} \) strictly inside the interval \((s, t)\) (not containing either endpoint \( s \) or \( t \)). Now we define the \( S(k, s, t) \) as a subset of \( I(k, s, t) \) of maximum cardinality such that no two edges in \( S(k, s, t) \) cross. Let \( A(k, s, t) := |S(k, s, t)| \) be its cardinality. Then there exists an ENCBM of size at most \( R \) if and only if \( A(2n, 0, 2n + 1) = n \).

We can calculate \( A(k, s, t) \) and \( S(k, s, t) \) using dynamic programming. Certainly, \( A(0, s, t) = 0 \) and \( S(0, s, t) = \emptyset \) for all \( s \) and \( t \). For the induction step, either \((i_{k+1}, j_{k+1}) \not\subset (s, t)\) or \( s < i_{k+1} < j_{k+1} < t \). If \((i_{k+1}, j_{k+1}) \not\subset (s, t)\), then clearly, \( A(k + 1, i, j) = A(k, i, j) \) and \( S(k + 1, s, t) = S(k, s, t) \). Otherwise, if \( s < i_{k+1} < j_{k+1} < t \), then \( S(k + 1, s, t) \) is either \( S(k, s, t) \) or \( \{i_{k+1}, j_{k+1}\} \cup S(k, s, i_{k+1}) \cup S(k, j_{k+1}) \cup S(k, j_{k+1}, t) \) and

\[
A(k, s, t) = \max\{A(k, s, t), 1 + A(k, s, i_{k+1}) + A(k, i_{k+1}, j_{k+1}) + S(k, j_{k+1}, t)\}. \tag{1}
\]

Since we must compute \( A(k, s, t) \) for all \( k \leq n^2 \) and \( s, t \leq n \), the above dynamic program can be computed in running time \( \mathcal{O}(n^4) \).

In the algorithm above, we do not use the fact that nodes are connected if they are within a distance \( R \) of each other; the above algorithm finds a maximal non-crossing set of edges of \( \text{any} \) graph whose vertices form the vertices of a convex polygon. The same result holds for points on the boundary of a simply connected polygon, if we impose the constraint that edges may not leave the polygon.
References


