OPEN PROBLEMS
33rd European Workshop on Computational Geometry (EuroCG 2017)
April 6, Malmö, Sweden
Introduction

Open Problems are a curious thing. My non-academic friends are always making fun when I tell them that I am looking for new problems. "Don’t you have enough problems?" "No, I need more!". But there are also more serious reasons that call for an open problem session. Imagine you are working on a very interesting research question and you could come up with a very nice, simple and elegant solution. You are very proud and send it to SoCG early December. By February you receive the reviews and you are very disappointed as the reviewers think that your result is really easy. This can happen and probably did happen many times. Unfortunately, results in theory are often judged by how complicated the proof was. But really, this is a very bad criteria. We should aim for interesting results regardless of the difficulty of the proof. Even more, we should consider simple proofs as an achievement.

Questions that are repeatedly asked by several people on open problem sessions and at other occasions clearly are interesting to the community. So we need these regular open problem sessions as an indicator to reviewers that might not be very familiar with the topic that they are reviewing that indeed people are interested in it. This is especially true to very hard problems with few papers on the topic.

Otherwise, sometimes you just desperately want to know the answer to some problem, regardless, on who is going to solve it. Posing open problems is also a great opportunity to find collaborators. Sometimes some people are afraid to share their best open problem as they fear that someone else might solve the question before them. It happened also to me that someone else solved a problem that I was working on before me. And yes it was frustrating at the time. Nevertheless, my personal opinion is that sharing leads to larger benefit for the community and also for each individual.

Tillmann Miltzow
1 Sum of Square Roots by Tillmann Miltzow

Before I present the problem itself, I want to give a disclaimer. This problem was asked before and may not be easy. But it lies at the core of computational geometry and deserves to be restated again. See The Open Problem Project for links to the literature and its origin [1].

If you ask a student in computational geometry if they can find the shortest path in a simple polygon, they can read the literature and find a relatively simple algorithm. If you ask the student or phd-student to give an algorithm that decides if the path is shorter than some integer $k$, then they might say this is very simple, as all they have to do is adding up the length of all the segments they constructed in the shortest path. Often even researchers are baffled, if you tell them that we do not know if this algorithm runs in polynomial time. Even more, we do not even know if the problem lies in NP. The curious fact that we can compute the path but not its length precisely is the topic of this Open Problem.

The underlying reason is that we do not know how to decide the following problem in polynomial or even non-deterministic polynomial time.

**Sum Of Square Root**

**Input:** $a_1, \ldots, a_n, k \in \mathbb{N}$

**Question:**

$$\sqrt{a_1} + \ldots + \sqrt{a_n} < k?$$

**Question 1.** Can the Sum Of Square Root problem be decided in polynomial time?

In practice the algorithm works for two reasons: First, we are often not in need of a completely precise answer. The computer rounds all roots for some amount of digits and then adds all these numbers up with some bounded precision and returns an answer. Second, often geometric input comes from measurement, which are influenced by noise and thus are not completely precise in the first place.

Another reason, for this surprise might be linked to the real-RAM model. In the real-RAM model we are allowed to add, multiply and compare any two real numbers. Often it is additionally assumed that one can also compute the square root of a number within $O(1)$-time. In computational geometry, we are so used to make this strong assumption that it is sometimes taken for granted. My main aim is to raise awareness again.

We do know that the problem is in the complexity class $\exists \mathbb{R}$. This complexity class consists of all problems that are polynomial time reducible to ETR. And ETR is the problem of deciding if an arbitrary set of polynomial equations and inequalities can be satisfied by some real numbers. The encoding is as follows:

$$\exists x_1, \ldots, x_n : x_1^2 = a_1 \land \ldots \land x_n^2 = a_n \land x_1 > 0 \land \ldots \land x_n > 0 \land \sum_{i=1}^{n} x_i < k.$$  

Usually, ETR allows only the numerical symbols 0, 1, but for the sake of simplicity, we do not explain here how to encode integers and instead refer to Matousek [2] for this subtlety. As we know that $\exists \mathbb{R} \subseteq \text{PSPACE}$, we can conclude that the Sum Of Square Root problem can be solved with polynomial space. As more and more problems in computational geometry are shown to be $\exists \mathbb{R}$-complete, it is not unreasonable to believe that this problem will be eventually added to the list of $\exists \mathbb{R}$-complete problems. However, there is a fundamental difference to the known complete problems. The reductions showing hardness usually imply a so called “universality” theorem. Here is meant that the topology of any semi-algebraic set can be translated into the
problem at hand [3]. This cannot be done here, as the solution space of the above formula is either empty or contains a single point. Furthermore, in many $\exists\mathbb{R}$-hard problems it is not so difficult to find examples that need double exponential precision for a solution to be represented. As for example for the recognition of segment intersection graphs the following theorem holds:

**Theorem 1.** For every sufficiently large $n$, there are $n$-vertex segment intersection graphs for which every segment representation with integral endpoints has coordinates doubly exponential in $n$, that is, with $2^{\Omega(n)}$ digits.

Also such a phenomenon is not observed yet, for the Sum Of Square Root problem. It would show that at least the naive algorithm mentioned above does not work. These kind of results rely on the ability to simulate repeated squaring or taking repeatedly square roots. While, we can take square roots here, we cannot do this in a repeated fashion.

At last I want to mention that also many other problems in computational geometry that need the real-RAM and are not known to be in NP, at least partially due to the Sum Of Square Root problem. Minimum weighted triangulation, shortest Euclidean TSP-tour, Minimum Euclidean Spanning Tree and Euclidean Steiner Tree to name a few famous examples.

**References**


2 Approximating the Maximum Independent Set of Line Segments by Anna Adamaszek

Consider the following problem. We are given a set of line segments \( S \) in the plane, and the goal is to find a maximum independent set within \( S \), i.e., maximum cardinality set \( S' \subset S \) s.t. any two line segments of \( S' \) are non-intersecting. See Figure 1 for an example.

![Figure 1: An independent set of line segments is pictured in red.](image1)

The problem is NP-hard [4], and the best known approximation ratio for it is \( n^\epsilon \) [3]. However, in quasi-polynomial time we can get a \((1 + \epsilon)\)-approximate solution [1]. That is an indication that it should be possible to obtain a better polynomial time approximation for the problem.

By losing a logarithmic factor in the approximation ratio, we can reduce the problem to the special case where all input line segments intersect the same horizontal line, see Figure 2 for an example. (The reduction is similar as used in [2] for the case of axis-parallel rectangles.) For this setting also no better approximation than \( n^\epsilon \) is known.

![Figure 2: An independent set of line segments is pictured in red. All input line segments intersect the blue line.](image2)

**Related problems:** We can consider a more general version of the problem, where each line segment has an associated weight, and the goal is to compute a maximum weight independent set.

We can also consider a simpler setting, where each line segment is either horizontal or vertical. I am not aware of any better approximation algorithm than the trivial 2-approximation (i.e., computing optimal solutions for all vertical segments, and for all horizontal segments, and outputting one of these two solutions) for this setting.
References


3 Bichromatic Triangles in Arrangements by Stefan Felsner

The study of cells in arrangements of lines and pseudolines has a long tradition. Already in 1826 F. Levi showed that an arrangement of $n$ pseudolines contains at least $n$ triangles. By now we know precise upper and lower bounds for the number of triangles of various types of arrangements, see [1].

Now suppose that the arrangement is bi-chromatic, i.e., the lines resp. pseudolines are colored red and blue such that there is at least one of each color. In this setting we can ask about colored cells. It is easy to see that it is possible that all triangles are bi-chromatic, i.e. each triangle is incident to a red and to a blue line. The question is whether there is always at least one bi-chromatic triangle. We conjecture that this is true.

In 2014 I could show that this is true for arrangements of lines. Alexander Pilz came up with the following lovely variant of the proof: Imagine the blue and red lines of the arrangement being drawn on two distinct overhead slides. Shift the slides against each other until the combinatorial type of the arrangement changes. This happens with a triangle mutation, i.e., when the lines of a triangle of the arrangement become concurrent. The lines that become concurrent must use both slides, hence, they form a bi-chromatic triangle of the arrangement.

Despite some efforts we have not yet been able to show the existence of bi-chromatic triangles for pseudolines, not even for simple arrangements of red and blue pseudolines.

Recently we introduced arrangements of approaching pseudolines. The motivation was that the above proof with the shifted slides can be used to show that bi-chromatic arrangements in this class contain bi-chromatic triangles. This work was presented at EuroCG 2017, see [2].

References


4 Triangle-Free Colorings of Segment Intersection Graphs by Bartosz Walczak

Given a finite family $S$ of straight-line segments in the plane, we construct the intersection graph $G$ of $S$ as follows: the vertices of $G$ are the segments in $S$ and the edges of $G$ are the pairs of segments in $S$ that intersect. A segment intersection graph is any graph that can obtained this way. It is known that there exist segment intersection graphs that are triangle-free (i.e. contain no triple of pairwise adjacent vertices) and still have arbitrarily large chromatic number [6, 7]. The current problem is about a variant of chromatic number, the triangle-free chromatic number, defined as the minimum number of colors in a coloring of the vertices of the graph that excludes monochromatic triples of pairwise adjacent vertices.

**Question 2.** *Is the following statement true: for every $k \geq 3$, there is a constant $\alpha_k$ such that every segment intersection graph with clique number (i.e. maximum size of a clique) at most $k$ has triangle-free chromatic number at most $\alpha_k$?*

I have no clue about which answer is more likely. On the one hand, it is shown in [3] that the answer is positive for intersection graphs of boundaries of axis-parallel rectangles that are clean (an additional technical condition), although the construction of triangle-free graphs with arbitrarily large chromatic number from [6, 7] also works for that class of graphs. This suggest that a similar distinction between the ordinary chromatic number and the triangle-free chromatic number might hold also for segment intersection graphs. On the other hand, it is also shown in [3] that the answer is negative for general string graphs (intersection graphs of arbitrary curves in the plane), although the construction witnessing that negative answer requires that curves can cross arbitrarily many times.

Question 2 is related to a so-called quasi-planar graph conjecture. A geometric graph is a graph drawn in the plane with edges represented by straight-line segments. Such a graph is $k$-quasi-planar if no $k$-tuple of its edges pairwise cross. Pach, Shahrokhi, and Szegedy [5] conjectured that for every $k \geq 2$, there is a constant $\beta_k$ such that every $k$-quasi-planar geometric graph with $n$ vertices has at most $\beta_k n$ edges. For $k = 2$, this conjecture asserts a well-known property of planar graphs. It is also known to be true for $k = 3$ [2, 4] and $k = 4$ [1], even in the more general setting of topological graphs (when the edges are drawn as arbitrary curves), but it remains open for $k \geq 5$. The best known upper bound on the number of edges in $k$-quasi-planar geometric graphs for $k \geq 5$ is $O_k(n \log n)$ [8]. Positive answer to Question 2 would imply a linear upper bound for all $k$, making use of the known linear upper bound for $k = 3$.

**References**


5 Object Visibility Graphs by Franz J. Brandenburg

We consider geometric objects in the plane that are closed disk homeomorphs. Special cases are convex objects, simple polygons, convex polygons, and points. Two non-intersecting geometric objects \( o \) and \( o' \) see one another if there is an unobstructed visibility line \( L(p, p') \) from a point \( p \) (on the boundary) of \( o \) to a point \( p' \) (on the boundary) of \( o' \). Then there is a visibility between \( o \) and \( o' \). How shall we define a graph?

- by intersection
- by touching
- by visibility

**Definition 2.** A graph \( G = (V, E) \) is called an object visibility graph if there is a set of non-intersecting objects so that there is a one-to-one correspondence between the sets of vertices and geometric objects and there is an edge \( \{u, v\} \) if and only if the objects \( o_u \) and \( o_v \) corresponding to \( u \) and \( v \) can see one another.

It can be shown that internally triangulated planar graphs, triangulated 1-planar graphs, and hole-free k-map graphs [2] are object visibility graphs. Also trees, triangulated outerplanar graphs, and triangulated outer 1-planar graphs are object visibility graphs, whereas \( k \)-cycles with \( k \geq 4 \) are not object visibility graphs. Clearly, every point visibility graph [3] is an object visibility graph. The recognition problem of string graphs is NP-complete [6] whereas touching graphs of circles are planar [4]. Recently, Cardinal and Hoffmann [1] proved that the recognition problem of point visibility graphs is complete for the complexity class \( \exists \mathbb{R} \).

**Question 3.** We ask the following questions regarding visibility graphs.

(i) Is the recognition problem of object visibility graphs complete for the complexity class \( \forall \exists \mathbb{R} \)?

(ii) Does every object visibility graph admit a representation as an object visibility graph of polygons?

(iii) Which other graph classes are object visibility graphs?

**References**


6 Minimum-link paths in 3D by Irina Kostitsyna

The minimum-link path problem asks to find an obstacle-avoiding path between two given points with the minimum number of turns. In 2D the problem is well studied and many polynomial time algorithms are known, including in various special cases, such as finding a min-link path inside a simple polygon or finding a path restricted to a certain set of directions [1].

In computational geometry the standard model of computation is the real RAM, which represents data as an infinite sequence of storage cells which can store arbitrary real numbers. This model is often preferred for its simplicity, however, in some cases its assumptions lead to unrealistic results. As Kahan and Snoeyink observed in [2]:

“If one considers bit complexity, [...] merely representing a minimum-link path may require a superquadratic number of bits.”

Thus, minimum-link path algorithms that are linear under the real RAM model are no longer linear if analyzed under different more “realistic” models of computation (such as, for example, the word RAM).

There is a tight bound of \( \Theta(n \log n) \) on the bit complexity of minimum-link path bends in 2D [3]. From this it follows that the minimum-link path problem in 2D is in NP.

Now let us consider the problem in 3D:

<table>
<thead>
<tr>
<th>Minimum-link path in 3D</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> a polyhedral domain and two points in it,</td>
</tr>
<tr>
<td><strong>Problem:</strong> connect the points by a polygonal path with minimum number of edges.</td>
</tr>
</tbody>
</table>

The question of determining its complexity was mentioned as one of the open problems in the Handbook of Discrete and Computational Geometry [1] and The Open Problem Project [4], but it dates back to 1990 [5] (and possibly even earlier). The problem was shown to be NP-hard in [3]. However it is still open if the problem is in NP. What is known regarding the bit complexity of a bend of a minimum-link path in 3D is its upper bound of \( O(9^n) \) bits. This all leads to the following open problem:

**Question 4.** What is the bit complexity of a bend of a minimum-link path in 3D?

References


7 Comp. Complexity of Area-Universality by Linda Kleist

This open question is the main contribution of the following paper/talk of EuroCG 2017[1].

We are interested in straight-line drawings of plane graphs realizing prescribed areas for the inner faces. A plane graph \( G' \) is a redrawing of a plane graph \( G \) if both graphs have the same face structure. Given a face area assignment of postive real numbers to all inner faces of \( G \), we are looking for a straight-line redrawing of \( G \) realizing the prescribed face areas. If for every face area assignment there exists a realizing redrawing of \( G \), then \( G \) is area-universal.

By Area Universality we denote the algorithmic problem of deciding if \( G \) is area-universal.

\[\begin{array}{|l|}
\hline
\textbf{Area Universality} \\
\textbf{Input:} Plane graph \( G = (V, E) \). \\
\textbf{Question:} Does there exist an area-realizing straight-line redrawing of \( G \), for every possible face area assignment? \\
\hline
\end{array}\]

Our question is:

**Question 5.** What is the computational complexity of Area Universality?

In our EuroCG paper [1], we introduce and motivate the natural complexity class \( \forall \exists \mathbb{R} \) and conjecture that Area Universality is \( \forall \exists \mathbb{R} \)-complete. We also show that some variants of Area Universality are Area Universality-complete.

**Conjecture 6.** Area-universality \( \forall \exists \mathbb{R} \)-complete!

Very surprisingly, already for small graphs it seems hard to decide if it is area-universal. To illustrate this we offer a super cool mug to the first person who answers the following question:

**Question 7.** Is the graph in Figure 3 area-universal?

![Figure 3: Is this graph area-universal? Win a mug! :)](image)

References

8 Do polygons respect triangles? by Michael Hoffmann

It is easy to see that for any given finite point set $Q \subset \mathbb{R}^2$ in general position\(^1\), we can build a simple polygon $P$ whose vertices are exactly the given points, that is, $V(P) = Q$. In that case, we say that $P$ respects $Q$. The situation gets much more interesting if we consider the given points to be vertices of obstacles that the polygon must not cross.

Consider a finite set $T$ of triangles in general position. A simple polygon $P$ respects $T$ if (i) $V(P) = V(T)$ (the vertices of $P$ are exactly the vertices of the given triangles) and (ii) every edge $uv$ of $P$ is either an edge of some triangle from $T$ or the relative interior $uv \setminus \{u, v\}$ is disjoint from all triangles in $T$. The left part of the figure below depicts a red polygon that respects the five gray triangles.

**Question 8.** Prove or disprove: For every finite set $T$ of pairwise disjoint triangles, there exists a simple polygon that respects $T$.

The corresponding statement is true for line segments [1, 3] instead of triangles and for axis-aligned rectangles [2]. But for general quadrilaterals the corresponding statement is false in general [2], even for squares.

The edges eligible to be used by a respecting polygon are the edges of the visibility graph of the obstacle vertices. Hence we may also phrase the question in a slightly weaker form.

**Question 9.** Is the visibility graph of the vertices of $n$ pairwise disjoint triangles in general position always Hamiltonian?

The right part of the figure above shows a counterexample with 13 squares whose vertex visibility graph is not Hamiltonian [1, 2]. It is not hard to show that the disjointness requirement in Question 8 and 9 is necessary.

**References**


---

\(^1\)In this problem we use a very weak notion of nondegeneracy: there exist at least three noncollinear points.
9 On a Threshold Graph Model for Complex Networks by Irene Sciriha

In the last twenty years, a number of different classes of graphs have been proposed to model real-world networks. The questions we ask are:

**Question 10.** Given a network $G$, is there a threshold graph that is sufficiently close to $G$ that it can be used instead of $G$ for the purposes of computing network statistics, parameters and spectral properties?

**Question 11.** How can computation time be reduced by drawing on the nice geometric properties of threshold graphs?

Real-world networks often display topological features that require exponential-time algorithms. On the other hand, threshold graphs display a nested split graph (NSG) structure that can reduce the complexity of algorithms to determine network invariants. The vertex set of a NSG can be partitioned into cliques and co-cliques that give an equitable vertex-partition [1]. The reason that NSGs are so nice to work with is that all connected NSGs on $n$ vertices correspond to a unique binary string of length $n - 2$. Many graph invariants can be computed directly from it. For instance the string 0110001101, of length 11, corresponds to the NSG, shown in Figure 4 on 13 vertices, with vertex partition 2,2,3,2,1,2, where the first and last parts of the 0–1 string are increased by one. The entries 0 in the string correspond to the addition of isolated vertices and the entries 1 to a dominating vertices in the construction of the NSG. An edge between two parts of the vertex partition indicates that each vertex of one part is adjacent to each vertex of the other part.

![Figure 4: The NSG with creation sequence 0110001101](image)

We are using Markov chains and simulated annealing to determine a NSG close enough to a given $G$. The problem centres on the best distance functions to use for reasonable computation times and reliable values of graph invariants of the NSG to which the process converges.

**References**