Complexity of Graph Drawing Problems in Relation to the Existential Theory of the Reals

Bachelor Thesis of

Nicholas Bieker

At the Department of Informatics
Institute of Theoretical Informatics

Reviewers:	Prof. Dr. Torsten Ueckerdt
	Prof. Dr. Peter Sanders
Advisors:	Paul Jungeblut

Time Period: 22nd April 2020 – 21st August 2020

Statement of Authorship

Ich versichere wahrheitsgemäß, die Arbeit selbstständig verfasst, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde sowie die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet zu haben.

Karlsruhe, August 20, 2020

Abstract

The Existential Theory of the Reals consists of true sentences of formulas of polynomial equations and inequalities over real variables that are existentially quantified. The corresponding decision problem ETR asks if a given formula of this structure is true. Similar to the relation between SAT and NP, the complexity class $\exists\mathbb{R}$ is defined as the problems that are polynomially transformable into ETR.

We first classify $\exists\mathbb{R}$ as a class inbetween NP and PSPACE and present a machine modell equivalent to $\exists\mathbb{R}$. Then we take a look at multiple $\exists\mathbb{R}$-complete variants of ETR that are commonly used as a basis for $\exists\mathbb{R}$-completeness proofs. We investigate many of these proofs for problems from a graph drawing background and find a framework that starts at an $\exists\mathbb{R}$-complete restriction of ETR called ETR-INV, or its planar variants.

After that, we apply this framework to conduct our own $\exists\mathbb{R}$ completeness proof for the problem DrawingOnSegments where we are given a graph G and an arrangement of segments and have to draw the graph on the segments in a planar way. Finally, we show $\exists\mathbb{R}$-membership for three more graph drawing problems. RAC-Drawing and α-CrossingAngle restrict the angles of crossing edges to 90 degrees and to minimum α, respectively, and AngularResolution that only allows angles of at least α between consecutive edges on their common vertex. We suspect that these problems are also $\exists\mathbb{R}$-complete.

Deutsche Zusammenfassung

In dieser Arbeit beschäftigen wir uns mit der Komplexitätsklasse $\exists\mathbb{R}$. Diese wird über die Existentielle Theorie der Reellen Zahlen definiert, welche eine Menge von wahren Formeln ist, diese bestehen aus logischen Verknüpfungen von Polynomgleichungen und Polynomungleichungen über reellen Variablen, wobei alle Variablen existentiell quantifiziert sind. Das zugehörige Entscheidungsproblem, das für eine Formel ϕ dieser Struktur entscheidet, ob ϕ wahr ist, heißt ETR. $\exists\mathbb{R}$ besteht dann aus allen Problemen, die in einer ETR-Formel in polynomieller Länge darstellbar sind.

Zunächst zeigen wir, dass $\exists\mathbb{R}$ zwischen NP und PSPACE eingeordnet werden kann. Dann stellen wir ein Maschinenmodell für $\exists\mathbb{R}$ vor und führen mehrere $\exists\mathbb{R}$-vollständige Varianten von ETR ein, die als Basis für Vollständigkeitsbeweise genutzt werden. Anschließend stellen wir einige dieser Beweise vor und finden für eine Art von Reduktionen ein Gerüst, das von ETR-INV und planaren Varianten des Problems ausgeht.

Dieses Gerüst benutzen wir dann selbst, um $\exists\mathbb{R}$-Vollständigkeit für das Problem DrawingOnSegments zu zeigen, wo wir einen gegebenen Graphen planar auf eine Menge von Segmenten zeichnen müssen. Außerdem zeigen wir für drei weitere Probleme die Zugehörigkeit zur Komplexitätsklasse $\exists\mathbb{R}$: Bei RAC-Drawing müssen wir für einen gegebenen Graphen entscheiden, ob er mit geraden Linien als Kanten gezeichnet werden kann, wobei Kreuzungen zwischen Kanten rechtwinklig sein müssen. Das Problem α-CrossingAngle verallgemeinert das und erlaubt minimalen Winkel α für Kreuzungen. Bei AngularResolution müssen stattdessen die Winkel zwischen adjazenten Kanten am gemeinsamen Endknoten einen Mindestwinkel α haben. Wir vermuten, dass diese Probleme auch $\exists\mathbb{R}$-vollständig sind.

Contents

1. Introduction

In Complexity Theory, the goal is to find problems of similar complexity and group those problems into complexity classes to find properties of the whole set of problems. Additionally, if two problems are polynomially equivalent, an algorithm that solves one of them in polynomial time directly leads to an algorithm for the other as the problems can be transformed into each other in polynomial time. The current order of the main complexity classes is P $\subseteq$ NP $\subseteq$ PSPACE $\subseteq$ EXP with no answer to the question if the inclusions are proper. In the case of NP $\neq$ PSPACE, this classification is not really sufficient as the gap between NP-completeness and PSPACE-completeness is quite large and there are many NP-hard problems with unclear relations of complexity. One approach to close that gap is the Polynomial Hierarchy, though the resulting complexity classes are very technical and non-inutitive. In this thesis, we explore another approach for mainly geometrical and topological problems: the complexity class $\exists\mathbb{R}$.

The Existential Theory of the Reals consists of formulas $\exists x_1, \ldots, x_n : p(x_1, \ldots, x_n)$, where p is a quantifier-free formula of polynomial equations and inequalities over real variables, that are feasible. The corresponding decision problem ETR asks if, given a formula ϕ of this structure, there are real numbers which satisfy ϕ. This problem was shown to be decidable by Tarski [Tar98], and later to be in PSPACE by Canny [Can88]. Schaefer used ETR to define a complexity class he called $\exists\mathbb{R}$ [Sch09], the relation between the two being the same as between NP and SAT as $\exists\mathbb{R}$ consists of all problems that are polynomially reducible to ETR. Our goal in this thesis is to explore $\exists\mathbb{R}$ and its complete problems and also show $\exists\mathbb{R}$-completeness for a new problem, DrawingOnSegments.

There has been a lot of research on this complexity class in the last years, mainly additional problems that are $\exists\mathbb{R}$-complete have been found. Schaefer started by introducing the class and completing $\exists\mathbb{R}$-completeness proofs for problems where there has been earlier research to indicate a relation to the Esixtential Theory of the Reals [Sch09]. Matousek also expanded his earlier research on intersection graphs to show $\exists\mathbb{R}$-completeness for RECOG(SEG), the problem of deciding whether a given graph is an intersection graph of segments [Mat14]. After that, multiple other researchers expanded $\exists\mathbb{R}$ by conducting their own reductions from the same starting problem, SimpleStretchability, where the input is a simple arrangement of pseudolines and the question is if there is an isomorphic arrangement of line segments. This problem has been proven to be equivalent to a variant of ETR by Mnëv [Mnë88] and Shor [Sho] way before $\exists\mathbb{R}$ as a complexity class had been established. These reductions go into different kinds of topics: topology and realizability of topological expressions [DGC99], game theory [SŠ17, BM16] and more geometric and graph drawing problems [Sch13, Eri19, Car15].

Abrahamsen et al. then developed a new way to carry out $\exists\mathbb{R}$-completeness proofs by introducing ETR-INV, a new variant of ETR, as a starting point [AM19]. They used ETR-INV and its planar variants to show $\exists\mathbb{R}$-completeness for the art gallery problem of finding a set of points within a simple polygon that guards the whole polygon [AAM18]. Additionally, they showed that GraphInPolygon, where we have to find a planar drawing of a graph inside a polygon with some vertices having fixed positions on the boundary of the polygon [LMM18], and PrescribedAreaPE, where we are given a planar graph with an area assignment a and have to find a planar drawing that respects a while some vertices have fixed positions [DKMR18], are also $\exists\mathbb{R}$-complete in the same manner. In the latter paper, they also generalize $\exists\mathbb{R}$ to a more general complexity class allowing also a layer of univeral quantifiers to the formula, $\forall\exists\mathbb{R}$, and compare that to the beginning of the polynomial hierarchy in classic complexity theory.

Our own contribution in this paper is to show $\exists\mathbb{R}$-completeness for DrawingOnSegments. In this problem, we are given a graph $G = (V, E)$ with a combinatorial embedding and an arrangement of line segments of two different kinds: Each vertex belongs to a vertex segment and has to be placed on that segment, additionally we have obstacle segments that edges cannot pass. The problem is to find a straight-line planar drawing of G where each vertex of V is placed on the corresponding segment and no edge passes an obstacle segment. We will show that DrawingOnSegments is $\exists\mathbb{R}$-complete.

Additionally, we show $\exists\mathbb{R}$-membership for three additional graph drawing problems with focus on drawing edges in certain angles: AngularResolution where the angles between consecutive edges on the same vertex have to be at least as big as the input α, and RAC-Drawing as well as α-CrossingAngle where the crossing angles are regulated. We also discuss first approaches for reductions to show $\exists\mathbb{R}$-completeness for these problems.

The thesis is structured in the following way: In Chapter 2, we introduce basic definitions from the areas complexity theory, graph theory and algebra that will be used throughout the thesis. In Chapter 3, we then formally define the Existential Theory of the Reals and the corresponding complexity class $\exists\mathbb{R}$ and show that $\exists\mathbb{R}$ lies between the known classes NP and PSPACE. We also find a machine modell for $\exists\mathbb{R}$ and introduce different variants of the problem ETR. We use these variants in Chapter 4 to present different proofs of $\exists\mathbb{R}$-completeness for geometrical and graph drawing problems that have been made in the past years. We generalize the reductions into different categories and explain a framework for the reductions from ETR-INV and its variants. In Chapter 5, we implement that framework to show $\exists\mathbb{R}$-completeness for DrawingOnSegments and look at different variants of the problem. Finally, in Chapter 6, we deal with the angular graph drawing problems and give formulas that prove their membership in $\exists\mathbb{R}$.

2. Preliminaries

In this chapter, we introduce some of the basic terms and definitions that are used throughout the thesis. We group them by their original topic, not in the order they are used in the thesis.

2.1 Complexity Theory

First, we recall a few basic terms from complexity theory, and also add specific definitions we need in Chapter 3. To understand some of the following definitions, we first introduce the *turing machine* as the basic way to define computations in our setting. A deterministic turing machine M consists of an infinite tape, a finite control section and a head that is on a specified position of the tape and can change the symbol on its position. Formally, M is a quintuple $(Q, \Gamma, \delta, q_0, F)$ with a finite set of states Q, a subset of final states F, a starting state q_0, the finite set of symbols that are allowed to be written on the tape Γ and the function that represents a computation step of the machine, $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, N, R\}$. At the start, the input is written on the tape and the head is on the first symbol. In every step, M reads the symbol on its head position, computes the new state according to δ, writes a new symbol and moves the head one to the left, one to the right or not at all. The computation ends if either a final state is reached or if M is in a state where it does not change its state, position and tape anymore. M decides a problem Π if M finishes the computation for every input and ends in a final state exactly when the input is a yes-instance of Π. A *nondeterministic turing machine* N has an additional computation phase where it can write an arbitrary word y left of the input onto the tape and then switches to the normal computation phase that works the same way as the deterministic one. N accepts an input x if a y exists so that the computation phase ends in a final state, y is called a *witness* for x.

A deterministic turing machine M has a *polynomial time complexity* if for each input I with length $|I|$ it decides in time $p(|I|)$ for a polynomial p if M accepts I. Similarly, M has *polynomial space complexity* if the amount of space that T needs in the computation is bounded by a polynomial $p(|I|)$. For a nondeterministic turing machine N, the definition changes to asking if, for each input I, there exists a witness y such that N needs polynomial time/space to decide if I is accepted when y is written on the tape in the first phase.

A *complexity class* is a set of problems that are equivalent in complexity in regards to some property, mostly time or space that is needed to solve the problem. Some complexity classes, like P and NP, are defined via a machine modell that can decide exactly the problems in the class, others are defined by a basic problem that has to be at least as complex as all the problems in the class (as ETR for $\exists\mathbb{R}$). To express that formally, we need a new terminology: A *polynomial reduction* is a function f that transforms instances I from one problem Π_1 into instances $f(I)$ of a problem Π_2 that fulfills the following properties: f has to be computable by a turing machine in polynomial time, and f has to map yes-instances onto yes-instances and no-instances onto no-instances. Using this term, we can define complete problems for a complexity class C: A problem Π is *C-hard* if for every $\Pi_1 \in C$ there is a polynomial reduction $f : \Pi_1 \to \Pi$. If additionally $\Pi \in C$ holds, Π is called *C-complete*.

As we need the complexity classes NP and PSPACE later in the thesis, we want to shortly define them here. *NP* is the complexity class that contains exactly the problems that are decidable in polynomial time by a nondeterministic turing machine. *PSPACE* consists of the problems that are solvable by a deterministic turing machine with polynomial space complexity. One NP-complete problem that we will need in the thesis is 3SAT:

3SAT:
Input: Pair (U, C) with variables U and clauses C that consist of exactly three literals from the variables in U.
Problem: Is there an assignement $f : U \to \{t, f\}$ that assignes boolean values to the variables in U such that in every clause $c \in C$ at least one of the literals is true?

We need a final definition for our machine modell for $\exists\mathbb{R}$ in Section 3.3: $\mathbb{R}^*$ is a set that consists of sequences of real numbers with finite length. For $x \in \mathbb{R}^*$, we will note the length of the sequence, in other terms the amount of real numbers in x, as $|x|$.

2.2 Graph Theory

In this thesis , we consider many problems that are related to graphs and graph drawing, so we want to clarify the way we use a few terms and notations throughout the thesis:

We use $[n] = \{1, ..., n\}$ to describe the set of integers from 1 to n. A *graph* is a pair (V, E) consisting of an *vertex set* $V = [n]$ and an *edge set* $E \subseteq \binom{[n]}{2}$ which means that E consists of subsets of $[n]$ of size two. For the edge set, we also set $m = |E|$.

A *graph drawing* is a concrete assignment of vertices and edges onto the plane $\mathbb{R}^2$, this means that we assign each vertex coordinates from $\mathbb{R}^2$ and assign each edge a Jordan curve that starts and ends in the endpoints of the edge. There are different properties a drawing can have: A *planar drawing* is a drawing of a graph G where no two edges intersect. In a *straight-line* drawing of a graph G, every edge is drawn as a line segment. In an *orthogonal drawing* of G, the edges only run horizontally or vertically, though they are allowed to have turns and switch directions. Those concepts are visualized in Figure 2.1. Note that we sometimes refer to straight-line drawings as *rectilinear drawings* for consistency reasons as it is used in the original papers in this way, although the term is somewhat non-intuitive for that.

A *planar graph* is a graph that has a planar graph drawing. A *plane graph* is a concrete drawing of a graph that has no intersections between two edges. Note that, for the purpose of this paper, we allow *degenerate* drawings, which means that we also include drawings that are limits of sequences of normal graph drawings. This results in two edges being able to run on top of each other. A region bounded by the edges of a plane graph is called a *face*. The *combinatorial embedding* of a planar graph dictates the general structure for the planar graph drawing by giving an order in which the edges are orientated around a vertex for every vertex of the graph.

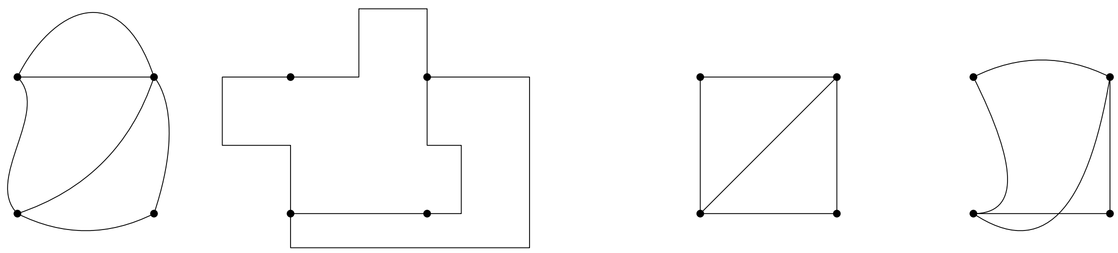


Figure 2.1: A planar drawing, an orthogonal drawing, a straight-line drawing and a non-planar drawing of a graph G

As a final definition, we introduce the concept of an *incidence graph* $G(\phi) = (V(\phi), E(\phi))$ to an ETR formula ϕ. $V(\phi)$ includes a vertex v_u for each variable u from ϕ, and a vertex v_p for each equation or inequality p in ϕ. There are no edges between two variable vertices and no edges between two equation vertices in $E(\phi)$, and the edge $v_u v_p$ is in $E(\phi)$ if and only if the variable u is used in p.

2.3 Algebra

Finally, in Chapter 3, we mention the connection between the Existential Theory of the Reals and its underlying algebraic relations. For that we need a few definitions:

A *semialgebraic set* S is a set of points in $\mathbb{R}^n$ that can be described via a quantifier-free formula ϕ of polynomial equations and inequalities. Each member $x \in S$ then has to be a solution of ϕ. If S can be defined by only conjunctions of polynomial equations and inequalities, S is called a *basic semialgebraic set*. If we additionally only allow equations, S is called an *algebraic set*.

A *real closed field* is a field F with the same first order properties as the real numbers: A sentence ϕ of first order logic is feasible with variables from F if and only if it is feasible with variables from $\mathbb{R}$.

To prepare for Mnëv's Universality Theorem, which we deal with in Chapter 4, we additionally need the concept of *Rank-3 Orientated Matroids*. As we do not work with them in detail in the thesis, we do not give an exact definition here and just intuitively explain the concept. A rank-3 oriented matroid is a combinatorial abstraction of a set P of points in the plane that uses the order type (defines for three points in a plane if they are collinear or form a left or a right turn) as a way to measure relations between three points. It can be expressed as a ternary predicate $\chi(p, q, r)$. This predicate has to fulfill the chirotope axioms cyclic symmetry, antisymmetry and Grassmann-Plücker relations. A more detailed explanation can be found here [CK13].

3. Complexity Class $\exists\mathbb{R}$

First, we define our main problem ETR and the corresponding complexity class $\exists\mathbb{R}$:

3.1 Definition

The Existential Theory of the Reals is the set of all true logic formulas from the existential first order logic over real numbers. These formulas include real variables x_i, which can be used in polynomial equations and inequalities with integer coefficients. The Existential Theory of the Reals then consists of all sentences of this structure that are feasible. This means that we can introduce all variables with existential quantifiers and comprise our definition of the Existential Theory of the Reals to all true formulas of the following structure: $\exists x_1, \dots, x_n : p(x_1, \dots, x_n)$, with p being such a logical combination of polynomial equations and inequalities. Formally, we can define the corresponding decision problem ETR in the following way:

ETR:
Input: Formula $\exists x_1, \dots, x_n : p(x_1, \dots, x_n)$ where p is a quantifier-free formula over the signature $\{0, 1, +, \cdot, <, \leq, =\}$ with connectives $\{\vee, \wedge, \neg\}$.
Problem: Are there real numbers $x_1, \dots, x_n$ for which the formula p is true?

An example for such an instance of ETR can be a simple formula of the following structure:

$$\phi \equiv \exists x, y, z \in \mathbb{R} : (x^2 + 3y = 1 \vee x = z) \wedge z \leq 0$$

This formula is feasible, e.g. with variables $x = z = -1, y = 0$, and therefore ϕ belongs to the Existential Theory of the Reals. Note that, while not technically allowed in the definition of ETR, we use multiple abbreviations such as writing x^2 instead of $x \cdot x$ and 3 instead of $(1 + 1 + 1)$, which are obviously equivalent. We use these abbreviations throughout the thesis.

Note that solution sets of the Existential Theory of the Reals directly correspond to the algebraic term semialgebraic set as their definitions are similar. This connection can be helpful when considering properties of semialgebraic sets that can be used in reductions in this topic. Mnëv [Mnë88], Schaefer [Sch09] and McDiarmid and Müller [MM13] all used this approach for their reductions, as we mention again later in the thesis. We do not go into detail for this direction though, as our thesis mainly looks at the reductions from a geometric viewpoint.

With the help of the problem ETR, we can now define a complexity class that includes all problems that can be expressed (in a formula of polynomial length) in the Existential Theory of the Reals:

Definition 3.1 ($\exists\mathbb{R}$)**.** *The complexity class $\exists\mathbb{R}$ consists of the problems that can be reduced to the Existential Theory of the Reals (ETR) in polynomial time.*

One can compare the relation of $\exists\mathbb{R}$ and ETR to the relation between NP and SAT. The biggest difference is that for SAT, NP-completeness was proven by Cook [Coo71] while for ETR, $\exists\mathbb{R}$-completeness follows from the definition of $\exists\mathbb{R}$. Still, both problems are the starting point for every hardness proof in NP and $\exists\mathbb{R}$ respectively. Known problems that are complete for $\exists\mathbb{R}$ generally are from topology, game theory, geometry or graph drawing. For this thesis, we focus on the graph drawing problems, for topology and game theory we refer to [SŠ17] and [BM16]. The complexity class was first proposed by Schaefer [Sch09], and then quickly adopted and expanded by other authors, notably Miltzow who, along with other colleagues, has designed reductions for multiple graph drawing problems which we explore later.

Note that, although the existential first order logic over real numbers is not expressable and computable by turing machines as the input is not finite, the input of the problem ETR (and thus of all problems in $\exists\mathbb{R}$ as they have to be transformable into ETR) is finite because only integer coefficiants are allowed. This means that we can use the same definition for polynomial transformations between problems in this context, although the solutions of the problems can contain real numbers.

We now classify our new complexity class into the hierarchy of known complexity classes and then discuss a theoretical equivalent machine modell to $\exists\mathbb{R}$ similar as turing machines to P and NP. After that, we introduce multiple variants of ETR that are equivalent in their complexity and will later be the foundation to show $\exists\mathbb{R}$-completeness for multiple geometrical and graph drawing problems.

3.2 Classification

While there have been successful attempts to generalize $\exists\mathbb{R}$ and to build a more detailed hierarchy by adding more quantifiers by Dobbins et al. ([DKMR18]), similar to the polynomial hierarchy, we focus on simply classifying $\exists\mathbb{R}$ itself. The result we get and explain in the following sections is:

Theorem 3.2 ([Sch09], [Can88])**.** *NP* $\subseteq \exists\mathbb{R} \subseteq$ *PSPACE*

3.2.1 NP $\subseteq \exists\mathbb{R}$

In this section, we prove that NP is a subset of $\exists\mathbb{R}$. For that, it is sufficient to show that a 3SAT instance can be polynomially reduced to an ETR-formula. 3SAT is NP-complete, so every problem in NP can be polynomially reduced to 3SAT and then to ETR, thus every problem in NP is also in $\exists\mathbb{R}$ and NP $\subseteq \exists\mathbb{R}$.

Note that is it unlikely that $NP = \exists\mathbb{R}$ holds as there are instances of many of the $\exists\mathbb{R}$-complete problems that require irrational coordinates for displaying their solutions, thus the solutions cannot be directly computed by turing machines. In Chapter 5, we explicitly give such an instance for the problem DrawingOnSegments.

Lemma 3.3 ([Sch09])**.** *NP* $\subseteq \exists\mathbb{R}$

Proof. We mainly follow the proof Schaefer gave in [Sch09], but modify it at a few points to make it easier to understand.

Let (U, C) be a 3SAT instance with variables U and C as the set of clauses. We construct an ETR-formula ϕ that is satisfiable if and only if (U, C) is satisfiable as well. First of all, for each 3SAT variable $u \in U$ we introduce a real variable x_u. For each of the real variables x_u also add the following formula: $(x_u = 1 \ \vee \ x_u = 0)$, all combined by conjunctions. Finally, let l be a literal in a clause $c \in C$. If l is not negated, use let $b_l = x_l$, if it is negated, then $b_l = (1 - x_l)$. For each $c \in C$ with literals x, y, z add $(b_x + b_y + b_z \geq 1)$ to ϕ.

This reduction is polynomial because for each 3SAT variable and each clause there is a constant part of ϕ, so the size of the formula is polynomial in the size of the 3SAT instance. If the 3SAT instance is satisfiable, an assignment of variables exists so that each clause is satisfied. For each 3SAT variable u that is true under the assignement choose $x_u = 1$, for each false variable $x_u = 0$ for the real counterpart. Every clause is satisfied, so in each of the $(b_x + b_y + b_z)$, one of the b_i has to be 1 so each of these formulas are true. That means the whole formula is satisfied. On the other hand, if ϕ is satisfied, there is an assignement that assigns each of the real variables x_u either the value 0 or 1. One can construct an assignment of 3SAT variables by setting u true exactly when $x_u = 1$, and false otherwise. Then, each of the clauses is satisfied because if there was a clause $x \vee y \vee z$ that was not satisfied, the formula $b_x + b_y + b_z \geq 1$ would not be satisfied as well and thus ϕ would not be true.

□

3.2.2 $\exists\mathbb{R} \subseteq$ PSPACE

For the other part of Theorem 3.2, we take a look at the development of algorithms that solve ETR throughout the later part of the 20th century. They develop from solving the more general first order logic over real numbers in exponential time and space to specific ETR-algorithms that only need polynomial space. We then shortly mention more modern and efficient algorithms and implementations to solve ETR. The running time of these algorithms is dependant on four variables: the number of variables n, the number of polynomials m, the total degree of the polynomials d and the coefficient bit length L.

Lemma 3.4 ([Can88])**.** $\exists\mathbb{R} \subseteq$ *PSPACE*

Historically, there were multiple different algorithms that can be used to solve the decidability of ETR-formulas. The first algorithms solve a more general problem, the general first order logic of the reals (which corresponds to real closed fields in an algebraic sense). The person who first showed that this problem is decidable was Alfred Tarski in 1948 [Tar98] where he used a method called quantifier elimination to reduce a formula of the first order logic to a formula that does not contain quantifiers and its satisfiability does not change. However, this algorithm is not practical because its time complexity is non-elementary and cannot be bounded by any tower of exponentials.

In 1975, John Collins designed a new algorithm to solve the same problem, quantifier elimination in the real first order logic [Col75]. He used a new technique called Cylindrical Algebraic Decomposition (CAD) to vastly speed up the computation time. Although the theoretical time complexity is double exponential, it is quite fast in practice and was implemented and improved multiple times since then, e.g. by Brown in the QEPCAD-B system [Bro03].

In 1987, Grigor'ev and Vorobjov designed an algorithm for solving ETR that reduced the complexity to $L(md)^{n^2}$, although still needing an exponential amount of space [GVJ88]. One year later, Canny developed an algorithm with polynomial space complexity for ETR, thus proving that $\exists\mathbb{R} \subseteq$ PSPACE [Can88]. From a complexity theory point of view, this result is important and has not been refined since. Renegar later improved Canny's algorithm to run in $L \cdot log(L) \cdot log(L)(md)^{O(n)}$ while the space complexitiy stayed polynomial, thus preserving a polynomial runtime if the number of variables is fixed, a property Canny's algorithm did not have [Ren88].

Although the algorithms of Grigorev and Renegar focus on a more specialized problem than Collins and have a better theoretical runtime, they are dramatically slower on small inputs as Hong discovered in 1991 [H+91]. While Collin's algorithm can solve inputs with small parameters in seconds on modern computers, Renegar und Grigorev would need more than a million years to calculate a solution even if every single parameter (number of variables, number of polynomials, total degree and coefficient bit length) is just 2. Because of that, these algorithms designed especially for ETR are mostly of theoretical value. Still, there have been attempts to at least use the new ideas for algorithms that perform in an acceptable manner for special cases of ETR [HRS93].

3.3 Machine Model

We present a machine modell for ∃ℝ similar to turing machines for P and NP. There have been attempts to conceptualize what machine modells for problems over real numbers could look like. The work we explore is from Blum, Shub and Smale and is called the BSS-machine [BSS+89]. While the BSS-machine itself corresponds to a wider field of problems, there are subclasses in the hierarchy of the BSS-machine that are achieved by removing features from the machine, and one of these subclasses directly corresponds to the Existential Theory of the Reals.

The BSS-machine is a register machine where it is assumed that real numbers can be saved, loaded and computed exactly in constant time and constant space (unlike reality where we would need to approximate them and run into space and time problems as well as rounding inaccuracies. Formally, the BSS-machine was defined in the following way by Grädel:

Definition 3.5 (BSS-machine [Gra07])**.** *A* BSS-machine B *over* $\mathbb{R}$ *is a register machine without an explicit working register. It consists of N instructions. The input $x \in \mathbb{R}^*$ is saved in the first $|x|$ registers. A configuration of B consists of 4 variables: the current instruction $k \in [0, \dots, N]$, the reading and writing registers $r, w \in \mathbb{N}$ and the current content of the registers, $z \in \mathbb{R}^*$. There are three types of instructions:*

- *Compute a rational function $f(z)$ and save the result in the first register, and optionally update the read and write registers to either the next one or the first one*
- *Branch if the first register has a non-negative content*
- *Copy the value of the read register to the write register*

Similar to the definitions of P and NP with turing machines, we can define $\mathrm{P}_\mathbb{R}$ and $\mathrm{NP}_\mathbb{R}$ in our BSS-setting. For $\mathrm{NP}_\mathbb{R}$ we do that explicitly because we need the class later:

Definition 3.6 ($\mathrm{NP}_\mathbb{R}$ [Gra07])**.** *A set $L \subseteq \mathbb{R}^*$ is in $NP_\mathbb{R}$ if there exists a nondeterministic BSS-machine B that decides L in polynomial time.*

We do not go into detail how a nondeterministic BSS-machine B can be defined. One way to look at it is analogously to nondeterministic turing machines: B can guess real numbers $y \in \mathbb{R}^*$ before the computation starts, y then acts as a witness that helps prove the correctness of the input x (if x is correct).

The BSS-machine was designed for a much broader context than the Existential Theory of the Reals, it can even be used for other rings than the ring of the real numbers. We can still find a subclass of this modell which corresponds to our problem, as Bürgisser and Cucker describe [BC06]. The subclass is denoted as $BP(NP^0_{\mathbb{R}})$. The two restrictions compared to the general class $NP_{\mathbb{R}}$ are that there are no registers allowed (indicated by the 0) and that the input has to be encoded as a string of 0 and 1 (indicated by the BP, short for boolean part). These restrictions narrow down the BSS-setting just enough that we land in our complexity class $\exists\mathbb{R}$:

Theorem 3.7 ([BC06])**.** *The class $BP(NP^0_{\mathbb{R}})$ of the BSS-machine modell directly corresponds to the Existential Theory of the Reals.*

For the proof, we need a variant of ETR called FEASIBILITY,which we reuse later in the thesis. This variant restricts ETR-formulas to only consist of one polynomial that is tested on its equality to zero:

FEASIBILITY:
Input: ETR instance, of the form $\exists x_1, \dots, x_n : g(x_1, \dots, x_n) = 0$.
Problem: Are there real numbers $x_1, \dots, x_n$ for which the formula is true?

Proof outline (Theorem 3.7). The main idea of the proof is to show that the problem FEASIBILITY is both complete for $\exists\mathbb{R}$ and for $BP(NP^0_{\mathbb{R}})$. We show in the next section that FEASIBILITY is indeed complete for $\exists\mathbb{R}$, although it restricts the structure of the formulas a lot. For now, we outline the proof by Bürgisser and Cucker [BC06] that FEASIBILITY, or $\text{FEASIBILITY}^{\mathbb{Z}}_{\mathbb{R}}$ (meaning that we have a polynomial over $\mathbb{R}$ with integer coefficients, exactly our setting) as it is denoted in their paper, is $BP(NP^0_{\mathbb{R}})$-complete. For that, they show that the more general problem $\text{FEASIBILITY}_{\mathbb{R}}$ where the coefficients can be real numbers as well is $\text{NP}_{\mathbb{R}}$-complete. This was in fact already shown by Blum et al. in their original paper [BSS+89] by reducing the $\text{FEASIBILITY}_{\mathbb{R}}$ to polynomials with a maximal degree of 4 and then proving $\text{NP}_{\mathbb{R}}$-completeness for that problem. Bürgisser and Cucker then outline that the reduction of an arbitrary problem in $\text{NP}_{\mathbb{R}}$ to $\text{FEASIBILITY}_{\mathbb{R}}$ directly leads to a reduction in the $BP(NP^0_{\mathbb{R}})$-context to FEASIBILITY, and consecutively, FEASIBILITY is indeed complete in $BP(NP^0_{\mathbb{R}})$. □

With this equivalence, we get a new possibility to define $\exists\mathbb{R}$, similar to how NP is canonically defined over nondeterministic turing machines. This topic has not yet been explored much though, and we are going to stop here as well as our main purpose of this thesis is to show how geometrical $\exists\mathbb{R}$-completeness proofs can be achieved. For that, we now lay the foundation by giving multiple possible starting points for these reductions.

3.4 Variants

Now we take a look at different versions of ETR that will be our toolbox to prove $\exists\mathbb{R}$-completeness for our different problems in the next chapters, for example FEASIBILITY. All these variants restrict the way the formulas and variables are constructed to allow for easier reductions.

3.4.1 Feasibility and StrictIneq

The first variant we explore is StrictIneq which only allows strict inequalities. Although these theories correspond to different algebraic constructs (as e.g. $x^2 = 2$ cannot be expressed without equations), the complexity classes one could define over them are identical, as Marcus Schaefer proved in [SŠ17].

Theorem 3.8 ([SŠ17])**.** ETR*,* Feasibility *and* StrictIneq *are polynomial-time equivalent.*

Proof outline. One direction of the proof is easy, because every operation allowed in StrictIneq is also allowed in ETR, so every problem portrayable in StrictIneqis trivially portrayable in ETR. To prove the opposite direction, Schaefer uses Feasibility of the previous section. He then proves that ETR reduces to Feasibility by using that every semi-algebraic set is a projection of an algebraic set, and proves that Feasibility reduces to StrictIneq by using multiple distance observations for semi-algebraic sets to transform the equation $g(x_1, ..., x_n) = 0$ into a system of inequalities, thus showing that ETR can be reduced to Feasibility which then can be reduced to StrictIneq. □

3.4.2 ETR-INV

While these variants already restrict ETR a lot, we go even further to ease our reductions even more. We limit the logical operations to just conjunctions, as well as having only equations of specific types. The problem ETR-INV does exactly that:

ETR-INV

ETR-INV:
Input: ETR instance where the equations and inequalities are all of one of the following forms: $x = 1, x + y = z$ or $x \cdot y = 1$. Additionally, the only logical operations that are allowed are conjunctions.
Problem: Are there real numbers $x_1, \ldots, x_n$ in $[0.5, 2]$ for which the formula is true?

This problem was introduced by Abrahamsen et al. [AAM18] to show that the art gallery problem[1] is ∃ℝ-complete and is the foundation of multiple similar reductions. Although it restricts the rules of ETR a lot, it is still ∃ℝ-complete:

Theorem 3.9 ([AAM18])**.** ETR-INV *is* ∃ℝ*-complete.*

Proof outline. First, ETR-INV is obviously in ∃ℝ as it is only a restriction of ETR and thus all allowed formulas in ETR-INV are allowed in ETR as well. For the ∃ℝ-completeness, we do not go into detail how the proof is done but instead refer to two papers where the proof is done in detail: In [AAM18], Abrahamsen et al. prove the ∃ℝ-completeness of ETR-INV to then use it as a basis to show ∃ℝ-completeness for the art gallery problem. They start with Feasibility and then slowly transform the formula into fitting into the allowed operations. In [AM19], Abrahamsen and Miltzow refine the work already done by them and others and give a more detailed and comprehensive proof by defining multiple other variants of ETR along the way. □

[1] As a reminder: The art gallery problem asks if, for a simply polygon P, there is a set of points within the polygon G that guards the whole polygon, in other words, for every point p on the boundary of P, there is a guard $g \in G$ so that the line segment pg is completely in the interior of P.

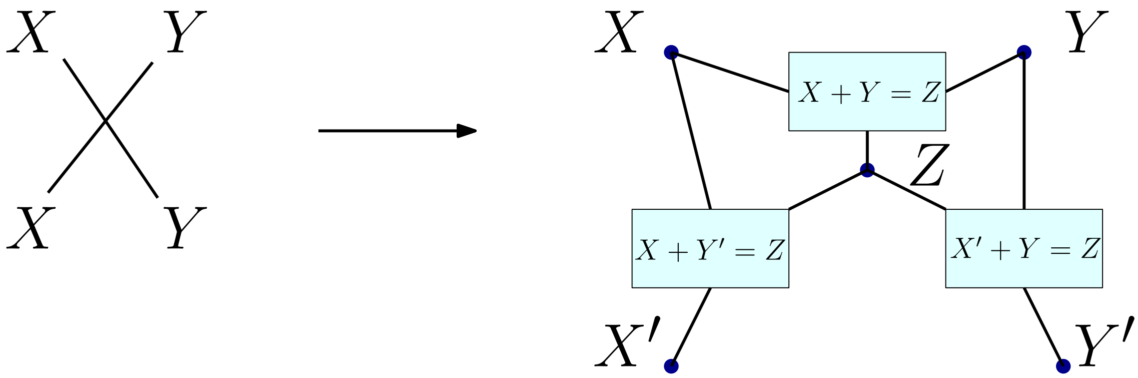


Figure 3.1: Crossing Gadget for Planar-ETR-INV (Original: [DKMR18])

While it is already easier to work with ETR-INV than ETR itself, we can restrict our formulas even more to even get planar incidence graphs. As a tradeoff, we get a polynomial increase in formula length and number of variables. These planar incidence graphs are the foundation of our own reductions for $\exists\mathbb{R}$-completeness later, so we introduce two versions of the Planar-ETR-INV problem, both used by Miltzow et al. [LMM18, DKMR18] in multiple reductions, which we cover in Section 4.3, and also by us in Chapter 5.

Planar-ETR-INV

This version of ETR-INV was used by Miltzow et al. [LMM18, DKMR18], as other problems need planarity in their reduction as opposed to the art gallery problem. Still, even with a planar incidence graph, the problem stays $\exists\mathbb{R}$-complete:

Planar-ETR-INV:
Input: ETR-INV instance where additionally the incidence graph of the formula is planar.
Problem: Are there real numbers $x_1, \ldots, x_n$ in the range $(0, 5)$ for which the formula is true?

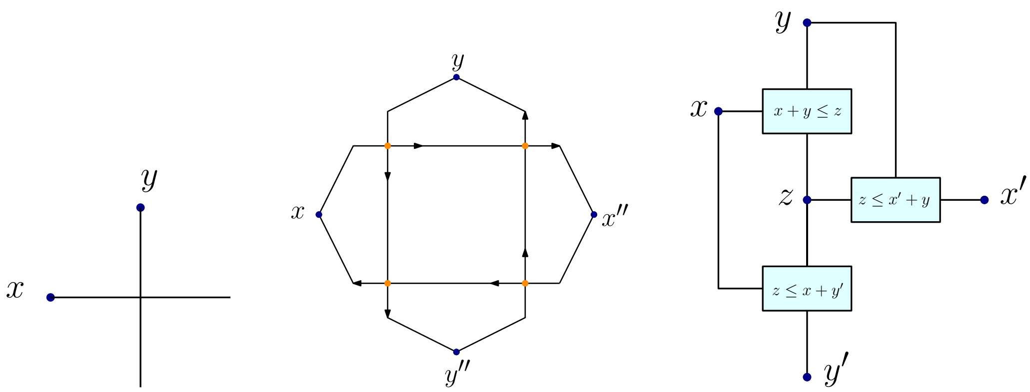
Theorem 3.10 ([DKMR18])**.** Planar-ETR-INV *is* $\exists\mathbb{R}$*-complete.*

Proof outline. We outline the proof of Dobbins et al. [DKMR18]. For that, we consider an ETR-INV instance I and take an incidence graph of I with minimal amount of crossings. We then systematically remove all the crossings by replacing them with a crossing gadget (Figure 3.1). This gadget introduces new variables X', Y', Z for the original variables X and Y and removes the crossing by adding addition gadgets that make sure that $X = X'$ and $Y = Y'$. In the end, we have no crossings left, and polynomially more variables and constraints as the gadget has constant size and there is at most a quadratic amount of crossings.

□

Planar-ETR-INV*

A few adjustments to the problem are beneficial for other kinds of reductions. We want to allow for inequalities instead of equations for addition and inversion, and also want to have a closed interval for our variables. This leads to the following version of Planar-ETR-INV, the distinction being signaled by the star in the problem name:



(a) Crossing between x and y

(b) High-level crossing gadget, orange dots are half-crossing gadget y

(c) A half-crossing gadget

Figure 3.2: Crossing Gadget for Planar-ETR-INV* (Original: [LMM18])

Planar-ETR-INV*:
Input: ETR instance where the equations and inequalities are all of one of the following forms: $x = 1, x + y \leq z, x + y \geq z, x \cdot y \leq 1, x \cdot y \geq 1$. Additionally, the only logical operations that are allowed are conjunctions and the incidence graph of the formula is planar.
Problem: Are there real numbers $x_1, \ldots, x_n$ in the range $[0.5, 4]$ for which the formula is true?

Theorem 3.11 ([LMM18])**.** Planar-ETR-INV* *is* ∃ℝ*-complete.*

Proof outline. This proof is similar to the one for Planar-ETR-INV, the only thing that changes is the structure of the crossing gadget (depicted in Figure 3.2). We cannot directly ensure equality anymore, that is why we need additional variables and 4 "half-crossing gadgets" (d) that only ensure $x \leq x'$ and $y \leq y'$. If we combine them in the way shown in (b) and add additional variables x'' and y'', we can ensure that $x = x''$ and $y = y''$ and eliminate the crossing.

□

4. Existing Problems and Reductions

After learning about the complexity class and the different variants of ETR, we now give an overview over the problems that are already known to be $\exists\mathbb{R}$-complete and categorize the corresponding reductions. We focus on geometrical graph problems, but there are other categories of problems that are known to be $\exists\mathbb{R}$-complete, too, like topological problems or problems belonging to game theory. All these reductions are from one of two problems: SimpleStretchability, which we cover in the next section, and ETR-INV (or its planar variants), which we already introduced in the last chapter. Figure 4.1 gives an overview over the reductions we look at in the following sections. We start with the proof by Mnëv that SimpleStretchaility is $\exists\mathbb{R}$-complete, and then generalize the reductions from the two basic problems.

4.1 Simple Stretchability of Pseudolines

The first problem to be proven $\exists\mathbb{R}$-complete was SimpleStretchability. For that, we first need a few definitions: A *pseudoline* is a simple curve in the plane. A *pseudoline arrangement* A is a set of pseudolines that is situated in the plane such that pairs of pseudolines cross each other exactly once. A is called *simple* if in each intersection no more than two pseudolines cross. Two pseudoline arrangements A and B are *isomorphic* if there exists a homeomorphism (continous bijective function with a continous reverse function) of the plane that maps A onto B. A *realization* of a pseudoline arrangement A is an arrangement of straight lines that is isomorphic to A. With these terms, we can define the problem:

SimpleStretchability:
Input: Simple arrangement A of pseudolines.
Problem: Does an arrangement of line segments exist that is isomorphic to A?

The proof of the $\exists\mathbb{R}$-completeness directly follows from Mnëv's Universality Theorem, which covers the reduction from StrictIneq to SimpleStretchability, but in a different context. We do not explain the concept of *stable equivalence* as it would need much more work and is not detrimental to our thesis, just note that it is a very strong statement that expresses much more than the reduction.

Theorem 4.1 (Universality Theorem [Mnë88])**.** *Every semialgebraic set is stably equivalent to the realization space of a rank-3 oriented matroid.*

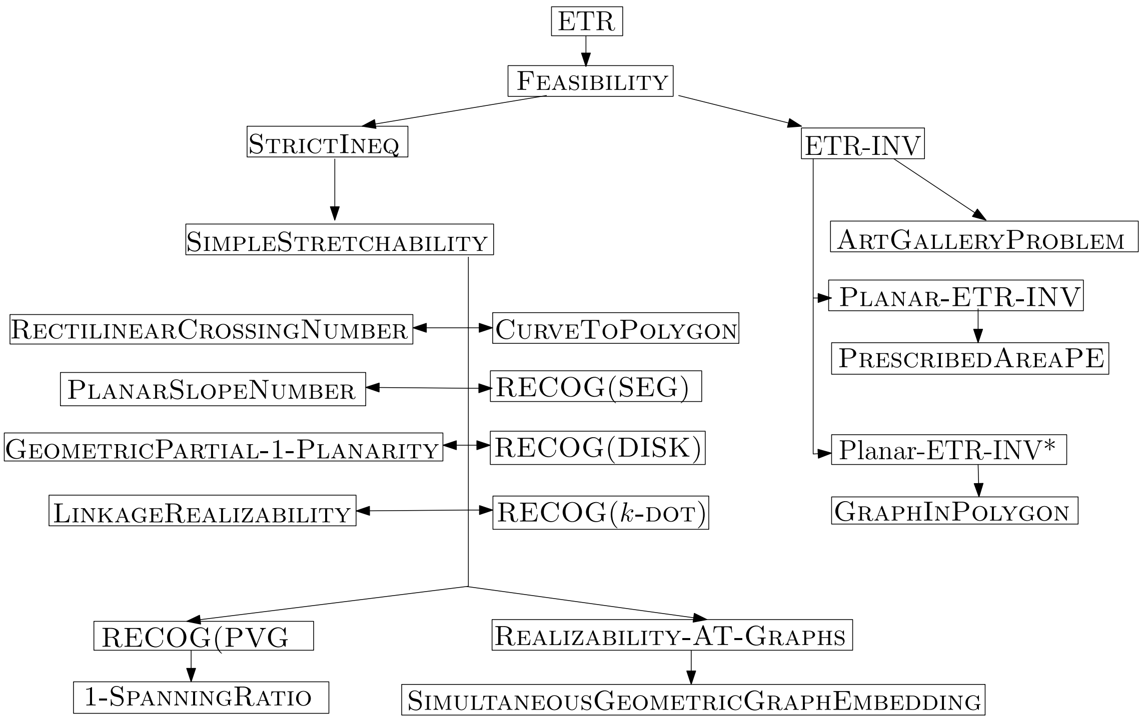


Figure 4.1: An overview over known ∃ℝ-complete problems, with the direction of the arrows indicating the direction of the reduction

We do not prove this theorem here because it is very technical and more algebraic, instead we refer to the original proof by Mnëv and the later work by Shor [Sho]. Shor simplified the proof and explicitly showed that the stretchability problem is NP-hard. Instead, we use the Universality Theorem to show that SimpleStretchability is ∃ℝ-complete. Indeed, the claim directly follows from the Universality Theorem as the oriented matroid can be seen as a pseudoline arrangement A and its realization space corresponds to a realization of A. Mnëv's proof can directly be used to get a procedure that transforms a StrictIneq instance into such a rank-3 orientated matroid, and thus into an instance of SimpleStretchability.

Theorem 4.2 ([Mnë88, Sho]). SimpleStretchability *is ∃ℝ-complete.*

Note that there are other approaches to prove this theorem, e.g. Matoušek [Mat14] used a geometrical reduction to show the statement rather than the algebraic approach by Mnëv and Shor.

Now that we have our first geometrical ∃ℝ-complete problem, we can take a look at the different kind of reductions that were made to other geometrical and graph drawing problems:

4.2 Reductions from SimpleStretchability

There are too many problems in this category to go into detail for each reduction, so we focus on two problems with different kinds of reductions: RectilinearCrossingNumber and CurveToPolygon. For the other problems, we give a problem definition and a short idea how the reduction is made.

4.2.1 Rectilinear Crossing Number

For the first problem of this kind, deciding whether a graph has a rectilinear crossing number of at least k, we give a full proof of ∃ℝ-completeness to give an indication as to how such a proof looks like. The *rectilinear crossing number* of a graph G, denoted as lin-cr(G), is the minimal number of crossings a straight-line drawing of G can have. The decision problem takes an integer k and asks for an input graph G if lin-cr$(G) \leq k$. This problem is $\exists\mathbb{R}$-complete even when G is restricted to be cubic, that means that every vertex in G has a degree of three.

RectilinearCrossingNumber:
Input: Cubic graph G, integer k.
Problem: Is there a straight-line drawing of G with at most k crossings?

Theorem 4.3 ([Bie91, Sch09])**.** RectilinearCrossingNumber *is* $\exists\mathbb{R}$*-complete.*

Proof. The proof of such a statement is done in two parts: First we show that the problem is in $\exists\mathbb{R}$ by giving an ETR-formula that is polynomial in size in the input size and is equivalent to the problem, and then we show that the problem is $\exists\mathbb{R}$-hard by reducing ETR to our problem. For RectilinearCrossingNumber, Schaefer proves that the problem is in $\exists\mathbb{R}$ [Sch09] while Bienstock gives a reduction from SimpleStretchability to RectilinearCrossingNumber [Bie91].

RectilinearCrossingNumber is in $\exists\mathbb{R}$

We need a few preliminaries to comprehend Schaefer's proof: Let $G = (V, E)$ be the input graph. We assume that $V = [n]$ and $E \subseteq \binom{[n]}{2}$. We assume that the edges of G are orientated in an arbitrary way such that we have a clearly defined head and tail for each edge. With these orientations, we define functions $h, t : E \to V$ that return the endpoints of an edge e. The formula is build by having a pair of variables (x_i, y_i) for each vertex encoding the position of the vertex in a planar drawing of G and then checking the properties of the problem for that drawing.

Schaefer's general idea for the formula is to allow at most k edges to cross by assigning additional variables z_{e_i,e_j} to each pair of edges (e_i, e_j) and to ensure that at most k of these z_{e_i,e_j} are greater than 0 which indicates that the pair of edges is allowed to cross. He also ensures that if z_{e_i,e_j} is not greater than 0, then the corresponding edges do not intersect. Note that we only need to check each pair of edges once, and that the same edge does not cross itself, so we only need the additional z_{e_i,e_j} for $i < j$.

In detail, he needs three predicates for that. The first predicate $\text{atmost}_k(z_{e_1,e_2}, ..., z_{e_{m-1},e_m})$ checks if at most k of the variables are greater than zero. The predicate $\overline{\text{collinear}}(x_1, y_1, x_2, y_2, x_3, y_3)$ ensures that the three points $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ are not collinear (do not lie on a line), and the final predicate $\overline{\text{cross}}(x_1, y_1, x_2, y_2, x'_1, y'_1, x'_2, y'_2)$ ensures that the corresponding line segments with endpoints (x_i, y_i) and (x'_i, y'_i) respectively do not intersect.

With those predicates, Schaefer builds the formula for checking if lin-cr(G) $\leq k$:

$$
\begin{aligned}
&\exists (x_1, y_1), \ldots, (x_n, y_n), z_{e_1,e_2}, \ldots, z_{e_{m-1},e_m} : atmost_k(z_{e_1,e_2}, \ldots, z_{e_{m-1},e_m}) \\
&\wedge \bigwedge_{i<j<k\in[n]} \overline{\text{collinear}}(x_i, y_i, x_j, y_j, x_k, y_k) \\
&\wedge \bigwedge_{e_i,e_j\in E, i<j} (z_{e_i,e_j} > 0 \vee \overline{\text{cross}}(x_{h(e_i)}, y_{h(e_i)}, x_{t(e_i)}, y_{t(e_i)}, x_{h(e_j)}, y_{h(e_j)}, x_{t(e_j)}, y_{t(e_j)}))
\end{aligned}
$$

We do not explain the predicates further as that would be very technical and add little value here. The implementations can be found in the original paper [Sch09].

Rectilinear Crossing Number is ∃ℝ-hard

We give Bienstock's reduction and then briefly justify its correctness. Bienstock starts with an instance of SimpleStretchability, an arrangement A of n pseudolines. He then gradually transforms that into a graph G along the following steps (depicted in Figure 4.2) and chooses $k = 5n(n-1)$:

1. Copy each pseudoline and place the copy directly next to the original segment.
2. Add vertices at each end of a pseudoline and each intersection between pseudolines to form a planar graph. For each original pseudoline $l \in A$, we call the four resulting vertices that represent the ends of the original pseudoline and its copy $u_{i,j}(l), i, j \in \{1, 2\}$ with i representing if the vertex belongs to the original pseudoline or the copy and j to which end of the pseudoline the vertex belongs.
3. For each original pseudoline l, add vertices $c_j(l)$ between the neighbouring end points of the original pseudolines and its copy $u_{1,j}(l)$ and $u_{2,j}(l)$ and edges between the neighbouring vertices on each end. Additionally, add edges e_l between the newly added vertices $v_1(l)$ and $v_2(l)$ belonging to the same line segment, and also add edges between neighbouring $u_{i,j}(l_k)$ to form a cycle on the boundary of the graph in the obvious way.
4. Copy the whole graph G_1, we call the copy G_2, add a matching between the corresponding vertices on the outer cycle.
5. Replace each edge in both of the cycles on the boundaries of G_1 and G_2 and the matching by $5n(n-1) + 1$ paths of length two where each path does not share its inner vertex or edges with any other path.

The resulting graph G has a crossing number of $5n(n-1)$ exactly when G is stretchable because every crossing has to be in the interior of the two smaller graphs G_1 and G_2 because of the way the matching and cycles are thickened. If A is stretchable, G can be drawn with $5n(n-1)$ crossings because we can draw G in the way depicted in Figure 4.2 and only have intersections between the edges e_l. Each pseudoline crosses each other exactly once and we have 5 crossings per intersection (both edges also cross the the two paths around the other edge), so we get $1/2 \cdot 5n(n-1)$ crossings per copy (as we have $1/2 \cdot n(n-1)$ pairs of pseudolines), and thus exactly $5n(n-1)$ crossings in general, thus lin-cr$(G) \leq k$. On the other hand, these crossings always have to exist in any straight-line drawing of G, so if there is a straight-line drawing of G with lin-cr$(G) \leq k$, the rest of the graph has to be plane, thus it automatically has the required structure and omits a realization of A. □

4.2.2 CurveToPolygon

The second problem in this section we take a closer look at is CurveToPolygon. This problem is different from the others as it is not a graph problem, but because it is different and the proof is quite intuitive and comparably easy to understand we include it here. The problem is defined in the following way:

CurveToPolygon:
Input: Self-intersecting closed curve γ, integer m.
Problem: Is there a polygon with m vertices that is isotopic to γ, in other words, can γ be continuously deformed into a polygon with at most m vertices without changing the pattern of intersections?

Erickson proved the problem to be ∃ℝ-complete:

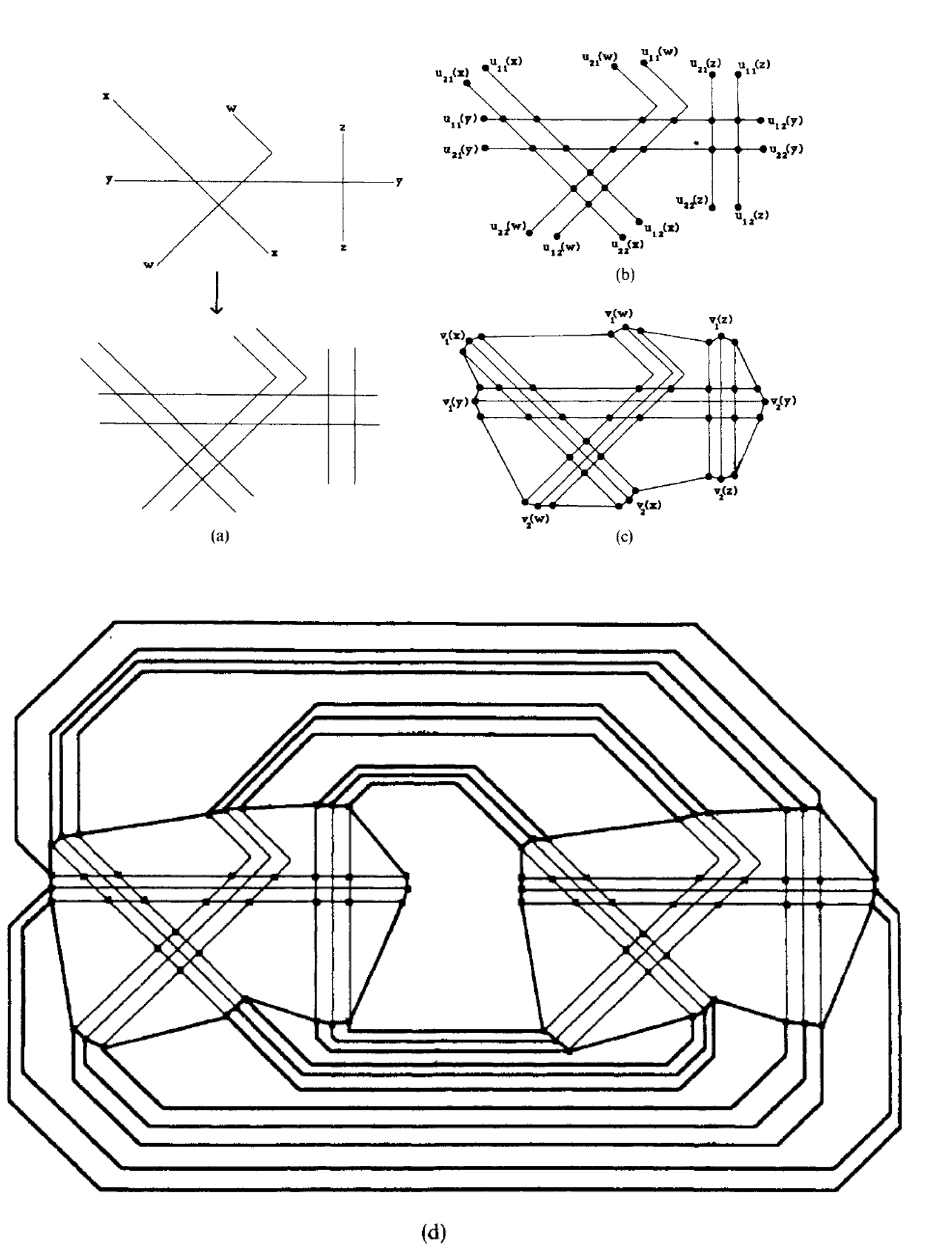


Figure 4.2: The steps of the reduction for RectilinearCrossingNumber (Source: [Bie91])

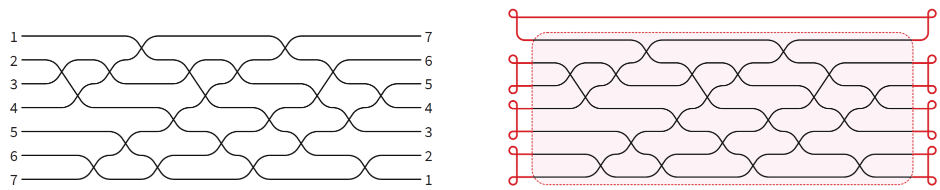


Figure 4.3: Pseudoline arrangement and the resulting CurveToPolygon instance (Source: [Eri19])

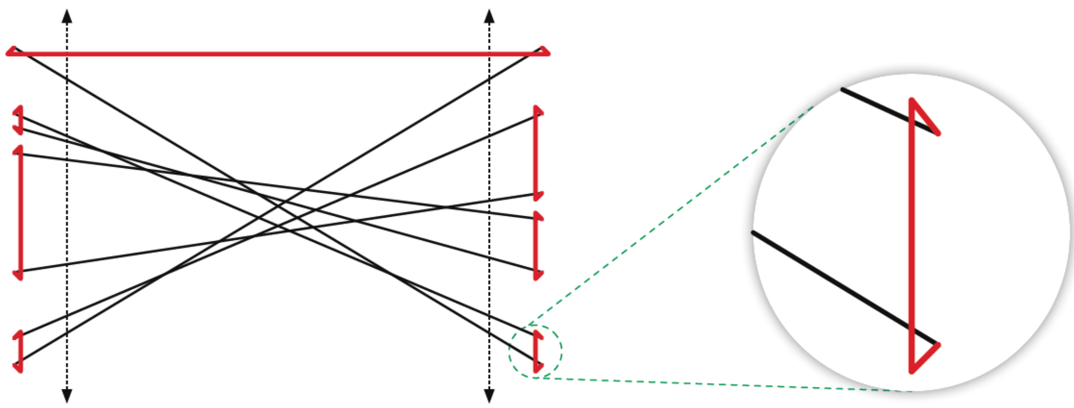


Figure 4.4: The resulting polygon that realizes the pseudoline arrangement (Source: [Eri19])

Theorem 4.4 ([Eri19])**.** CurveToPolygon *is* $\exists\mathbb{R}$*-complete.*

Let A be a pseudoline arrangement with n pseudolines. Erickson assumes the pseudolines to be arranged in a specific way as depicted in Figure 4.3. He then extracts a closed curve γ by connecting neighbouring line segments in the way depicted in Figure 4.3. This figure shows the case if there is an odd number of line segments; then the top ends on both sides are connected. He also sets the parameter m to be four times the number of pseudolines.

This was the whole reduction, and directly corresponds to the stretchability problem. The vertices of the polygon have to be in the added end loops, as each loop needs two vertices to be isotopic to a polygon. This means that there can be no additional vertices inside the closed curve where the original pseudolines where, so if there is a polygon with m vertices that is isotopic to γ, its edges that do not connect vertices from the same end gadget directly correspond to a straight-line realization of the set of pseudolines (as depicted in Figure 4.4).

4.2.3 Other Problems

We have gone into detail for two of the reductions, but there are more problems that are $\exists\mathbb{R}$-complete and have been shown to be that way by reduction from SimpleStretchability. We shortly define the problems, give a general idea of the proof and link the papers where the reductions are done in detail. Note that, as it is our main focus in this thesis, we try to give a complete overview over problems that deal with graph drawings and have omitted a few geometrical problems that do also belong in this category. For a few more of those, we refer to [Car15].

Intersection Graphs of Segments

For a set of segments S, an *intersection graph* $G = (V, E)$ is a graph with a vertex v for each segment l_v with the property that $uv \in E$ if and only if l_u and l_v intersect. The

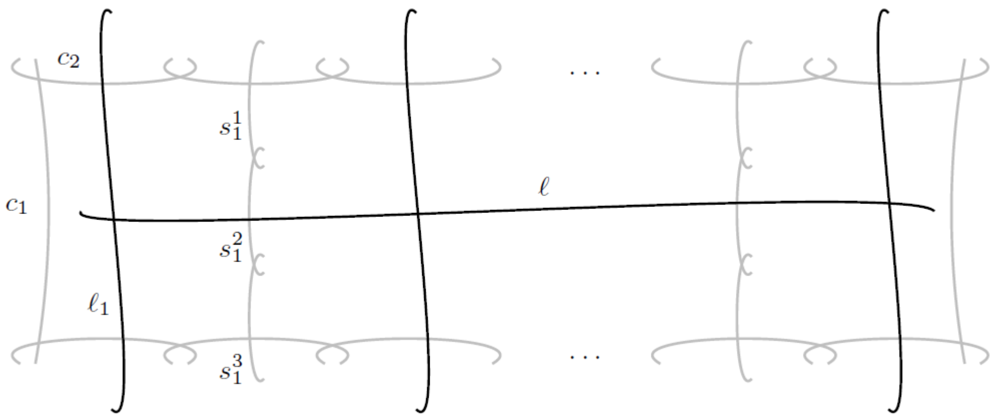


Figure 4.5: The ordering gadget for RECOG(SEG) (Source: [Sch09])

class of all graphs that are an intersection graph for a set of segments is SEG, the decision problem RECOG(SEG) asks if a given abstract graph G is part of SEG:

RECOG(SEG):
Input: Abstract graph $G = (V, E)$.
Problem: Is there a set of segments S such that G is the intersection graph of S?

Matoušek and Schaefer showed that this problem is $\exists\mathbb{R}$-complete:

Theorem 4.5 ([Mat14, Sch09])**.** *RECOG(SEG) is $\exists\mathbb{R}$-complete.*

First, they add a triangle of segments that surrounds every intersection of a pseudoline arrangement A and cut all the segments shortly outside the triangle. Then, for each intersection, ordering gadgets are added (structure in Figure 4.5). They define G as the intersection graph of this arrangement. The ordering gadgets force the order in which the original segments intersect to be either in original order or exactly reversed in every line segment arrangement for G, thus the arrangement is stretchable if and only if G is in SEG.

Unit Disk Graphs and k-dot Product Graphs

Another $\exists\mathbb{R}$-complete recognition problem deals with the recognition of disk intersection graphs. A *unit disk graph* is an intersection graph corresponding to disks in the plane with an uniform radius. The corresponding decision problem is RECOG(DISK):

RECOG(DISK):
Input: A graph G.
Problem: Is there an arrangement of disks with uniform radius in the plane such that G is the corresponding disk intersection graph?

McDiarmid and Müller indirectly showed that this problem is $\exists\mathbb{R}$-complete as they concentrated on integer representations of the problem. Because of that, the reduction is very technical and not geometrical, so we do not give an overview how the reduction is done.

Theorem 4.6 ([MM13])**.** RECOG(DISK) *is $\exists\mathbb{R}$-complete.*

Note that, as a generalization of that, Kang and Müller show that the problem stays $\exists\mathbb{R}$-complete if we move to other dimensions. The decision problem for these k-dot product graphs is defined in the following way:

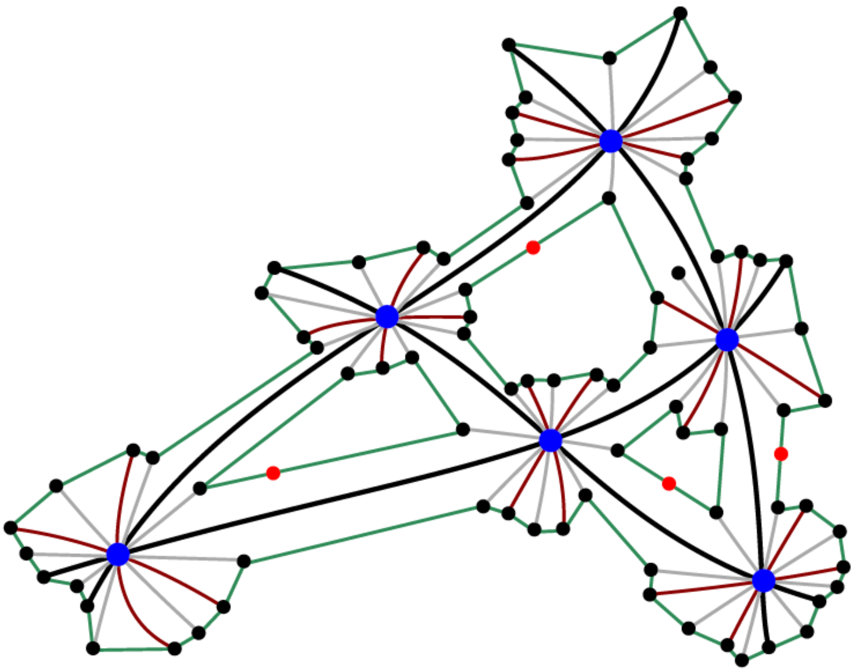


Figure 4.6: The planar graph G, original line segments are black, intersection vertices blue (Source: [Hof17])

RECOG(K-DOT):
Input: Graph $G = (V, E)$, integer k.
Problem: Is there a subset S of points in $\mathbb{R}^k$ such that each vertex $i \in V$ is identified with a point s_i and an edge ij exists if and only if the scalar product $s_i \cdot s_j$ is at least one?

Planar Slope Number

The *planar slope number* of a planar graph G is the minimal amount of different slopes of edges in a plane straight-line drawing of G. The corresponding problem of deciding whether a graph with maximum degree Δ has planar slope number $\Delta/2$ is $\exists\mathbb{R}$-complete:

PLANARSLOPENUMBER:
Input: Planar graph G with maximum degree Δ.
Problem: Is there a plane straight-line drawing of G with at most $\Delta/2$ different slopes?

Theorem 4.7 ([Hof17])**.** PLANARSLOPENUMBER *is* $\exists\mathbb{R}$*-complete.*

Hoffmann takes a pseudoline arrangement A of n pseudolines and constructs a planar graph G with $\Delta = 2n$ where each vertex that represents an intersection has degree Δ (G is pictured in Figure 4.6). G is drawable with n slopes if and only if A is stretchable and a drawing of G realizes L because in a graph with slope number $\Delta/2$ opposite edges of a vertex with degree Δ have the same slope, and G is constructed in a way that the corresponding parts of a pseudoline on an intersection vertex are on the opposing side.

Point Visibility Graphs

As recognition of certain types of graphs seems to be of equal complexity for many different of these types, we now consider another recognition problem. This time, we have to decide if a given graph G is a *point visibility graph*, which is a graph $G = (V, E)$ that corresponds to a set of points P in the plane with the property that $uv \in E$ if and only if the points p_u

and p_v see each other, which means that there is no other point $p \in P$ on the line segment between p_u and p_v. Cardinal and Hoffman show that this decision problem is $\exists\mathbb{R}$-complete:

RECOG(PVG):
Input: Graph $G = (V, E)$.
Problem: Is G a point visibility graph, in other words, does a set P with $|P| = n$ of points in the plane and a bijective function between P and V exist so that $uv \in E$ if and only if p_u and p_v see each other?

Theorem 4.8 ([CH17])**.** RECOG(PVG) *is* $\exists\mathbb{R}$*-complete.*

We want to highlight that Cardinal and Hoffman use a different method to prove $\exists\mathbb{R}$-completeness for this problem as they followed the idea of Mnëv and showed that a semialgebraic set is stably equivalent to the realization space of an instance of RECOG(PVG).

There is a direct relation between point visibility graphs and the spanning ratio of graphs. The *spanning ratio* of a graph $G = (V, E)$ is the minimal k so that, for each pair $u, v \in V$ the edge distance on the shortest path between u and v is at most $k \cdot ||uv||$. As Aichholzer et al. showed [ABB$^+$20], the answer to the question if a given graph has a spanning ratio of one is yes if and only if the graph is a point visibility graph. Thus, the following problem is also $\exists\mathbb{R}$-complete:

1-SpanningRatio:
Input: Graph $G = (V, E)$.
Problem: Can G be drawn as a proper straight-line drawing with spanning ratio 1?

Realization of AT graphs and Simultaneous Geometric Embedding

An *abstract topological graph* is a pair $AT = (G, R)$ with R being a set of pairs of edges from G. R expresses which edges of G are allowed to cross in a drawing of G. A *weak realization* of AT is a drawing of G where only the edge pairs in R cross, if all of them cross in the drawing, it is just called a *realization.* For a given AT-graph, deciding whether it is (weak) realizable with a straight-line drawing is $\exists\mathbb{R}$-complete, as shown by Kynčl. We are going to focus on the weak version of the problem as it leads to another $\exists\mathbb{R}$-complete problem:

WeakRectilinearRealizability:
Input: Abstract topological graph $T = (G, R)$.
Problem: Does a straight-line drawing of G exist where only edge pairs from R cross?

Theorem 4.9 ([Kyn11])**.** WeakRectilinearRealizability *is* $\exists\mathbb{R}$*-complete.*

Kynčl constructs an AT-graph T from a pseudoline arrangement A with n pseudolines by placing a circle around all the intersections in A. He then defines T as a cycle of $6n$ vertices with n chords corresponding to the pseudolines and $2n$ additional paths that connect certain vertices of the cycle and have length $n - 1$. Two types of crossings are allowed: chords can cross each other and the edge p_j of a path j can cross the chord c_j. T is realizable if and only if A is stretchable as the chords directly correspond to the stretched pseudolines.

Cardinal then used that problem to show $\exists\mathbb{R}$-completeness for another graph drawing problem, Simultaneous Geometric Embedding (or k-SGE) as it is equivalent [Car15]. Note that the reduction only works when k is for some constant c at least in $\Omega(n^c)$ for the size n of the original instance.

k-SGE:
Input: k Graphs $G_1 = (V, E_1), \ldots G_k = (V, E_k)$ with a shared vertex set V.
Problem: Does a set of coordinates $P \subset \mathbb{R}^2$ and a bijection $\pi : V \to P$ exist such that every induced drawing of any G_i is planar?

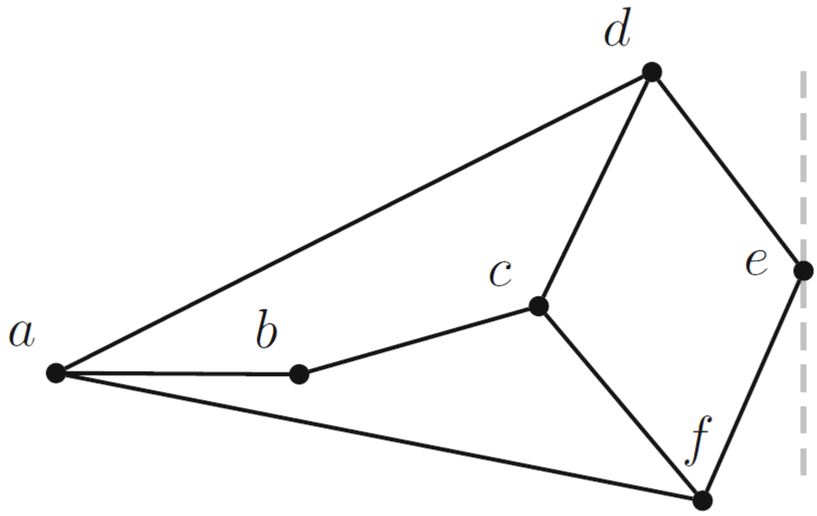


Figure 4.7: A Peaucellier linkage for LinkageRealizability (Source: [Sch13])

Linkage Realization and Unit Distance Graphs

Another graph drawing problem proven to be $\exists\mathbb{R}$-complete by Schaefer is the problem of drawing a graph with straight lines and given lengths for each of the edges:

LinkageRealizability:
Input: Graph $G = (V, E)$, function $l : E \to \mathbb{R}_{>0}$.
Problem: Is there a straight-line drawing of G where every edge e has its corresponding length $l(e)$?

Schaefer showed that this problem is $\exists\mathbb{R}$-complete even when we consider unit lengths:

Theorem 4.10 ([Sch13])**.** LinkageRealizability *is* $\exists\mathbb{R}$*-complete.*

The idea of the reduction is to use Peaucellier linkages (Figure 4.7) to make sure that three points have to be drawn on a line. These Peaucellier gadgets can be built in a way that the edges are uniform. Schaefer transforms a pseudoline arrangement A into a graph G by adding vertices on each intersection and end of a pseudoline and then replace edges between consecutive vertices on the pseudolines by the Peaucellier gadgets. G then is realizable if and only if the A is stretchable as ensured by the Peaucellier gadgets.

Partial Geometric 1-Planarity

Schaefer also researched the problem PartialPlanarity where for a graph G and a subset of edges F the graph is to be drawn in such a way that the edges of F have no intersections. Without further restrictions, Schaefer showed that the problem is solvable in polynomial time. He did not find a classification for the problem if the edges should be drawn as straight lines though. However, he showed $\exists\mathbb{R}$-completeness for a slight variant:

PartialGeometric-1-Planarity:
Input: Graph $G = (V, E)$, subset of edges $F \subseteq E$.
Problem: Is there a straight-line drawing of G such that every edge $e \in F$ has at most one crossing?

Theorem 4.11 ([Sch14])**.** PartialGeometric-1-Planarity *is* $\exists\mathbb{R}$*-complete.*

For the reduction, Schaefer surrounds the intersections of the pseudoline arrangement A with a parabola R. He then constructs a graph from that by playing vertices on each intersection of a line segment with the boundary, on each inner face of the resulting arrangement and an additional one below the parabola. He adds edges corresponding to the dual graph of the pseudoline arrangement and K_6 gadgets to ensure that a drawing of the graph leads to a realization of A.

4.3 Reductions from ETR-INV and its Variants

Overall, as we mentioned before, the reductions from SimpleStretchability differ very much and depend on the right idea for the specific problem. Now, we consider the second approach for showing $\exists\mathbb{R}$-completeness, reductions from ETR-INV, where we find a structure in the reductions that is much easier to replicate than the SimpleStretchability proofs. As a reminder, here is the general ETR-INV problem:

ETR-INV:
Input: ETR instance where the variables are equations and inequalities are all of one of the following forms: $x = 1, x + y = z, x \cdot y = 1$. Additionally, the only logical operations that are allowed are conjunctions.
Problem: Are there real numbers in $[0.5, 2]$ for which the formula is true?

The reductions from ETR-INV, as opposed to the ones in the previous section, all follow the same blueprint: We build specific structures in the setting of our problem to encode variables and to implement the different kinds of allowed formulas. We also need a way to transfer values of variables. We give three examples where this general idea is implemented in different ways. As a side note, Abrahamsen et al. [AMS20] modified this approach to also work for geometrical packing problems instead of graph drawing problems.

4.3.1 Drawing a Graph in a Polygonal Region

The problem we mainly focus on is GraphInPolygon. We reuse and slightly modify most of the gadgets later in Chapter 5 for our own proof. The problem is the following:

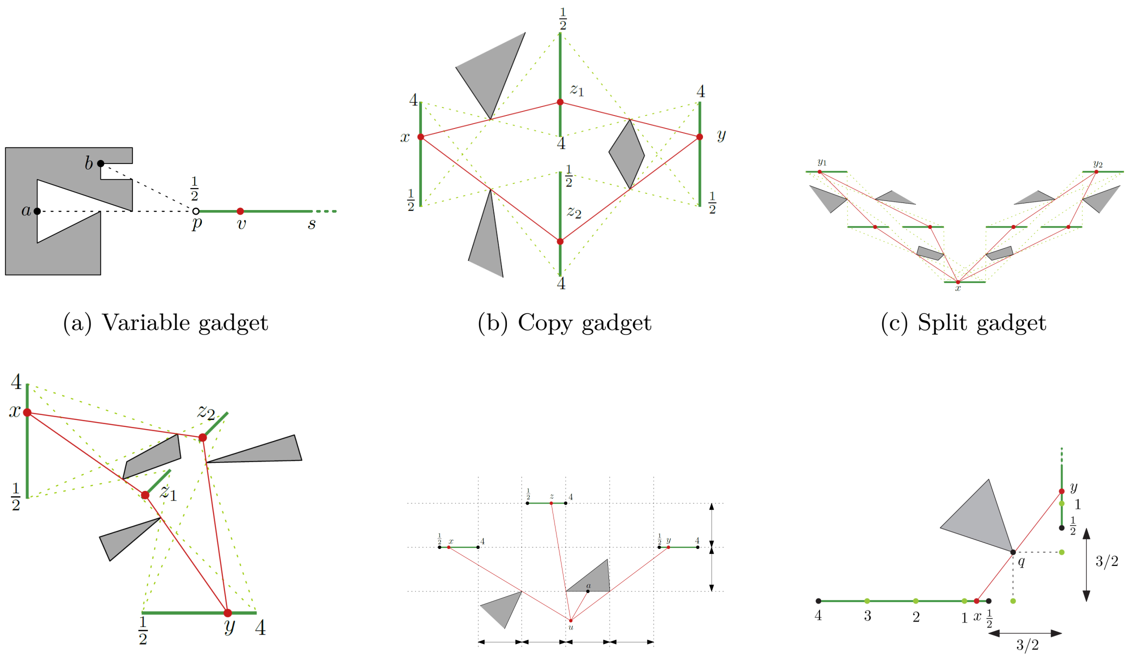
GraphInPolygon:
Input: Planar graph $G = (V, E)$, polygonal region R, subset of V with fixed positions on the border of R.
Problem: Is there a planar straight-line drawing of G where the fixed vertices are drawn on their assigned positions?

Lubiw et al. [LMM18] show that this problem is complete in $\exists\mathbb{R}$, using the blueprint for reductions from ETR-INV problems:

Theorem 4.12 ([LMM18])**.** GraphInPolygon *is* $\exists\mathbb{R}$*-complete.*

The general idea of the proof is to take an instance I of Planar-ETR-INV*, compute an orthogonal straight-line drawing of the incidence graph $G(I)$ of I and transform $G(I)$ into an instance of GraphInPolygon. For that, we need geometrical figures to express the equations and inequalities from the ETR-INV-formulas in terms of our problem, we call them gadgets in the rest of the thesis. We need gadgets for expressing variables and their values, and we need gadgets for the different kinds of formulas that Planar-ETR-INV* allows. We also need additional gadgets for transforming edges of the incidence graph into our problem instance. In the following paragraph, we introduce and explain those different kinds of gadgets (depicted in Figure 4.8).

First of all, we need a variable gadget to represent the variables and encode their values in our drawing. For that, we force a vertex onto a line segment and use the position of the



(a) Variable gadget (b) Copy gadget (c) Split gadget

(d) Turn gadget (e) Addition gadget (f) Inversion gadget

Figure 4.8: The gadgets for GraphInPolygon (Source: [LMM18])

vertex on the line segment to represent the value of the variable. How that is done can be seen in Figure 4.8 a): The fixed vertex a and the two peaks of the polygon ensure that the vertex p has to be at the same height as a, and the vertex b forces v to be on the right side of p. On the other side of the gadget, there is a mirrored construction that forces v to also be to the left of s, thus v has to be situated on the line segment ps.

We can divide the other gadgets into two categories: Formula gadgets that implement the different operations allowed in Planar-ETR-INV* formulas and transport gadgets that we use to replace the edges in the incidence graph to carry values of variables through the drawing. We have three different transport gadgets. The copy gadget just copies the value of a variable through space and replaces a straight-line part of an edge of the incidence graph. The turn gadget implements rotations of an edge of 90 degrees. The splitter gadget is there to transfer the value of a variable to different parts of the incidence graph if a variable vertex has a degree of more than one. Additionally, we have addition gadgets that ensure that a variable gadget z has value at most $x + y$ (or at least $x + y$ respectively) and inversion gadgets who ensure that, for two variable gadgets x and y, $x \cdot y \leq 1$ or $x \cdot y \geq 1$ holds respectively. The gadgets are depicted in Figure 4.8. We explain how and why they work in Chapter 5.

The final step now is to transform the planar incidence graph into an instance of GraphInPolygon. For that, we replace variable vertices with variable gadgets, formula vertices with the corresponding gadgets and edges with copy and turn segments. We again go into detail in Chapter 5 why this transformation is correct and always possible.

4.3.2 Art Gallery Problem

The art gallery problem is the first problem that was shown to be $\exists\mathbb{R}$-complete via reduction from ETR-INV. Abrahamsen et. al [AAM18] explicitly designed the ETR-INV problem to restrict the possible equations in the ETR formula to be better expressable with specific gadgets. The gadgets are built by using properties of the art gallery problem, which is defined in the following way:

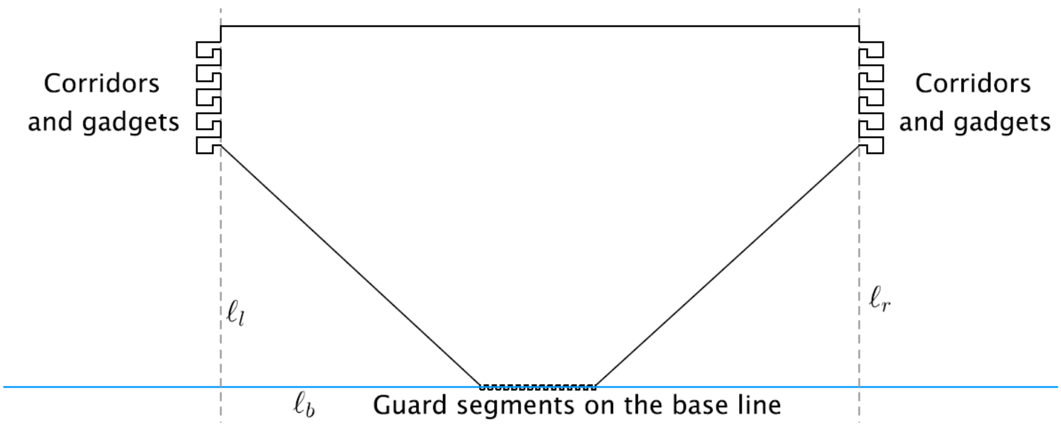


Figure 4.9: High-level polygon of the art gallery problem (Source: [AAM18])

ArtGalleryProblem:
Input: Simple polygon P with corners at rational coordinates, integer k.
Problem: Is there a set G of k guards (points in P) that guards all of P, so that for each $p \in P$ there is a $g \in G$ such that the line segment pg is completely in the interior of P?

Theorem 4.13 ([AAM18])**.** *The art gallery problem is $\exists\mathbb{R}$-complete.*

Lubiw et al. base the reduction on the general ETR-INV problem. They again construct gadgets for each type of equation in ETR-INV, but do not need the planar incidence graph that is the base of the previous reduction as the values of the variables are transmitted to the gadgets without additional constructions.

The high level sketch of the polygon can be seen in Figure 4.9. On the bottom, there are the guard segments which represent the variables of the formula. The gadgets are on the left and right side, with corridors at the beginning copying the values of the guard segments into the segments. The idea of the gadgets is that only specific points of the guard segments can see the entire gadget, and thus, to have an optimal set of guards, these points have to be included and the values of the variables can be manipulated in this way. For more details we refer to the original paper. For better understanding how the polygon can be used to manipulate values of variables, we discuss the guard segments a bit more.

As seen in Figure 4.10, we force a guard to be on a line segment s. For that, we cut two triangles on the left and right side into the polygon, so that a guard can only see the whole left triangle if it is at least at the height of s, and only sees the whole right triangle if it is at most at the height of s. Additionally, the ground is manipulated so that a guard on the height of s can only see everything below it if it is exactly in the boundaries of s. The reduction manipulates the number of allowed guards in a way that only one guard can be spared to guard this area, so the guard has to be on s. If we force a specific value for the variable, we can add more notches so that the position of the guard on s is even more restricted, e.g. to the value one as seen in the right side of Figure 4.10.

4.3.3 Prescribed Area PE

The last problem that belongs in this category that we briefly mention is PrescribedAreaPE (PE is short for partial extension) where we have to draw a plane graph with assigned areas for the facets and fixed positions for a subset of vertices:

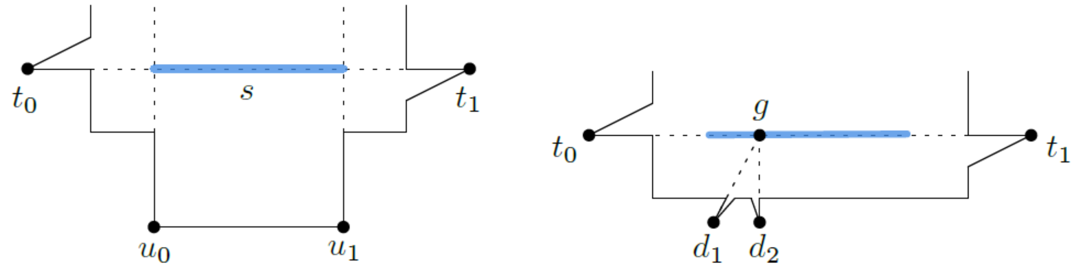


Figure 4.10: Guard segments for the art gallery problem (Source: [AAM18])

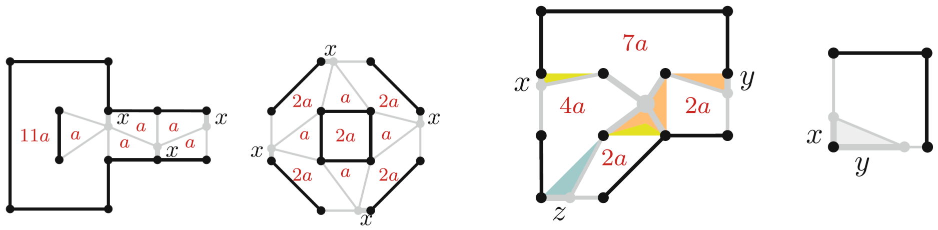


Figure 4.11: The gadgets for Prescribed Area PE (Source: [DKMR18])

PrescribedAreaPE:
Input: Planar graph $G = (V, E)$, vertices $V_f \subseteq V$ with fixed positions in the plane, fixed combinatorial embedding, function $a : F \to \mathbb{R}_{>0}$ that assigns an area to each face.
Problem: Is there a planar drawing of G that respects the combinatorial embedding and the fixed positions and such that the area of each face is exactly as a prescribes?

Theorem 4.14 ([DKMR18])**.** PrescribedAreaPE *is* $\exists\mathbb{R}$*-complete.*

The reduction is made from Planar-ETR-INV, the outline of the proof is similar to the one for GraphInPolygon. Only the construction of the gadgets is different. The gadgets follow the idea that again certain not fixed vertices are forced onto a line segment to encode the value of variable. The main manipulator this time is the area of the neighbouring faces, they can only match their supposed area if the variables have a specific value. In Figure 4.11 we show a few of them visually: On the left a variable gadget and with the beginning of a wire, then a splitter gadget, an addition gadget and an inversion gadget.

For more information we refer to the original paper. As a an additional note, in the same paper Dobbins et al. [DKMR18] also showed $\exists\mathbb{R}$-completeness for a more generalized version of the problem, Prescribed Volume, where the constructs are assigned to $\mathbb{R}^3$.

From now on, we use these observations to classify problems that were formerly not associated with $\exists\mathbb{R}$. We mainly use reductions from ETR-INV and its variants because they are more uniform and easier to imitate, but reductions from SimpleStretchability could also be possible.

5. Drawing a Graph on Segments

Now, we use the framework from the last section to show $\exists\mathbb{R}$-completeness for the following problem (Visualization in Figure 5.1):

DrawingGraphOnSegments:
Input: Graph $G = (V, E)$, set $S = S_v \dot{\cup} S_o$ of segments with vertex segments S_v that are closed and pervious and obstacle segments S_o that are open and impenetrable, bijective function $f : V \to S_v$, fixed combinatorial embedding.
Problem: Is there a straight-line planar drawing of G that fulfills the embedding so that each vertex v is located on the corresponding segment $f(v)$ and that the edges do not cross the obstacle segments?

To show that this problem is $\exists\mathbb{R}$-complete, we first give an equivalent formula to show $\exists\mathbb{R}$-membership, and then prove that it is also $\exists\mathbb{R}$-hard. Afterwards, we discuss alterations to the problem and their impact on the complexity of the problem.

5.1 $\exists\mathbb{R}$-Membership

As the input of the problem, we have the graph $G = (V, E)$, the set $S = S_v \dot{\cup} S_o$ of line segments and a bijective function $f : V \to S_v$. We need a few definitions to construct an ETR-formula for our problem. Let $V = [n]$ as well as $E \subseteq \binom{[n]}{2}$. We assume that the edges of the graph are orientated in an arbitrary way, so that we get $h, t : E \to V$ as functions that return the head and tail vertices of an edge respectively, and we assume to have start and end functions for the segments which return the coordinates of the endpoints of the line segment (more detailed: startX and startY for the x and y value of the coordinates, and endX and endY for the other endpoint). We enumerate the vertex segments in a way that s_i is the corresponding segment for vertex i according to f. We also need two functions c_h and c_t that return the clockwise neighbour of every edge according to the embedding for the head and tail vertex.

For the formula, we need pairs of variables to encode the vertices and we need to check four things to ensure that the drawing fulfills the requirements of the problem: Are all the vertices on the corresponding vertex segment, is the drawing of the graph planar, do the edges not intersect with the obstacle segments and is the combinatorial embedding fulfilled. We split these four checks into smaller formulas and combine them with conjunctions to form our ETR formula. Thus, the general outline of the formula is the following:

$$\exists (x_1, y_1), ..., (x_n, y_n), r_1, .., r_n, (s_{1,2}, t_{1,2}), .., (s_{m-1,m}, t_{m-1,m}) :$$
$$\text{OnSegment} \wedge \text{Planar} \wedge \text{NoIntersection} \wedge \text{Embedding}$$

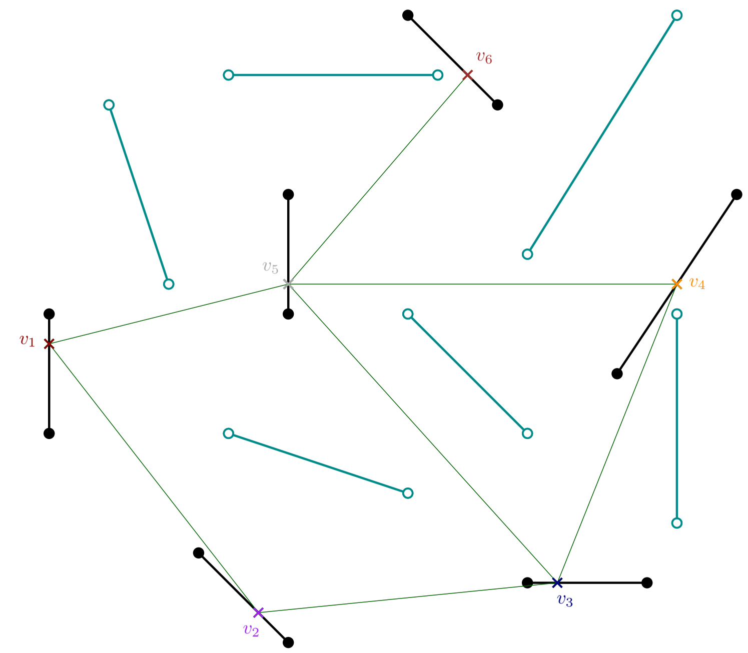


Figure 5.1: An instance with blue obstacle segments, black vertex segments and green edges

Here, the (x_i, y_i) variables encode the positions of the vertices, while the other variables are auxiliary variables for the different parts of the formula. We have a polynomial amount of variables, so the reduction works in polynomial time as each part has polynomial length.

Note that in this and any further $\exists\mathbb{R}$-membership constructions, to counteract possible divide-by-zero problems, we assume that no segment or edge has slope ∞ or $-\infty$. We can do that because if we have an edge or segment with that slope, we can turn our whole drawing clockwise until no infinite slope is left (which will happen because we only have a finite amount of edges/segments). Only the relative positions of our objects to each other are relevant to our problems, not the absolute positions in the plane, and we do not change those.

OnSegment:
To check if all vertices are on their respective vertex segments, we need to transform the vertex segment s_i into a parametrized depiction of the form $z = a + r(b - a)$ for both x and y coordinates and then check if the point is on the segment. For that, we construct a system of equations with two equations and one parameter r_i. If there is an r_i that fulfills both equations and lies on the segment itself and not just on the line through the segment, which corresponds to checking if r_i is between 0 and 1 (0 and 1 included as the segment is closed), then the vertex lies on the segment (see Figure 5.2).

$$\text{OnSegment} \equiv \bigwedge_{s_i \in S_v} (x_i = \mathit{startX}(s_i) - r_i \cdot (\mathit{endX}(s_i) - \mathit{startX}(s_i))$$
$$\wedge\, y_i = \mathit{startY}(s_i) - r_i \cdot (\mathit{endY}(s_i) - \mathit{startY}(s_i)) \wedge (0 \le r_i \wedge r_i \le 1)$$

Planar:
To check planarity, we need to compare all possible pairs of edges and determine whether they intersect in a point other than a common end point. For that, either the edges can be parallel (note that we allow degenerate planar drawings so parallel edges are allowed in all cases) or not parallel but also not crossing. The general formula is this:

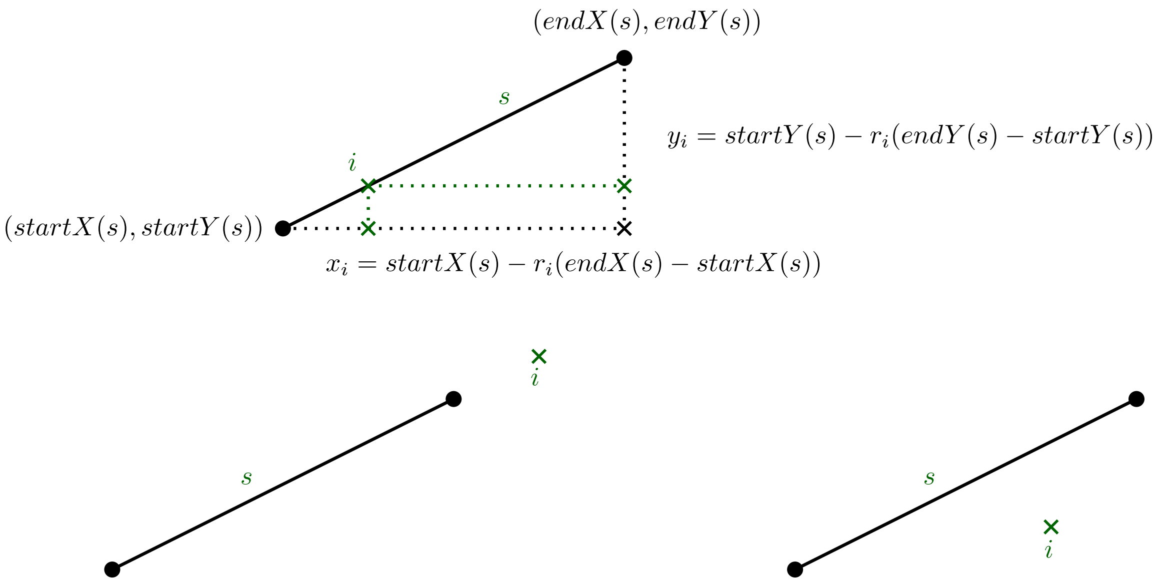


Figure 5.2: Visualization of the idea behind the formula OnSegment. Top: r_i exists and is between 0 and 1, bottom left: r_i exists, but is greater than 1, bottom right: r_i does not exist

$$\text{Planar} \equiv \bigwedge_{i,j,i<j \in [m]} \text{Parallel}(e_i, e_j) \vee \text{NoCross}(e_i, e_j)$$

Parallel:
Two edges are parallel if they have the same slope. The slope of an edge e can be computed by $(y_{t(e)} - y_{h(e)})/(x_{t(e)} - x_{h(e)})$. Note that, as no edges are vertical, we are not dividing by zero here. Rearranging the equation, we check if the slopes of both edges e_i and e_j are the same with the following equation:

$$\text{Parallel}(e_i, e_j) \equiv (y_{t(e_i)} - y_{h(e_i)})(x_{t(e_j)} - x_{h(e_j)}) = (y_{t(e_j)} - y_{h(e_j)})(x_{t(e_i)} - x_{h(e_i)})$$

NoCross:
To check if two edges cross we need to check where the straight lines through the edges cross (which we now assume to be non-parallel so the intersection exists). In order to do that, we parametrize the two edges to get an equation for the straight line: $(x(s_{e_i,e_j}), y(s_{e_i,e_j})) = (x_{h(e_i)} + s_{e_i,e_j}(x_{t(e_i)} - x_{h(e_i)}), y_{h(e_i)} + s_{e_i,e_j}(y_{t(e_i)} - y_{h(e_i)}))$, same for edge e_j with parameter t_{e_i,e_j}. If we now equate the two lines to get the point where they intersect encoded in $(s_{e_i,e_j}, t_{e_i,e_j})$, we get the following two equations, combined with a conjunction, which we call LineIntersection because we need them again in Chapter 6 (visualized in Figure 5.3):

$$\begin{aligned} \text{LineIntersection}(e_i, e_j) \equiv\ & s_{e_i,e_j}(x_{t(e_i)} - x_{h(e_i)}) - t_{e_i,e_j}(x_{t(e_j)} - x_{h(e_j)}) = x_{h(e_j)} - x_{h(e_i)} \\ \wedge\ & s_{e_i,e_j}(y_{t(e_i)} - y_{h(e_i)}) - s_{e_i,e_j}(y_{t(e_j)} - y_{h(e_j)}) = y_{h(e_j)} - y_{h(e_i)} \end{aligned}$$

These equations ensure that the point where the two lines intersect is encoded in the point $(s_{e_i,e_j}, t_{e_i,e_j})$. The two edges cross if the intersection lies on both edges, which translates to both coordinates being between 0 and 1.If one of the parameters is not between 0 and 1, the edges do not cross, which leads us to the following formula:

$$\text{NoCross}(e_i, e_j) \equiv \text{LineIntersection}(e_i, e_j) \wedge (s_{e_i,e_j} \leq 0 \vee s_{e_i,e_j} \geq 1 \vee t_{e_i,e_j} \leq 0 \vee t_{e_i,e_j} \geq 1)$$

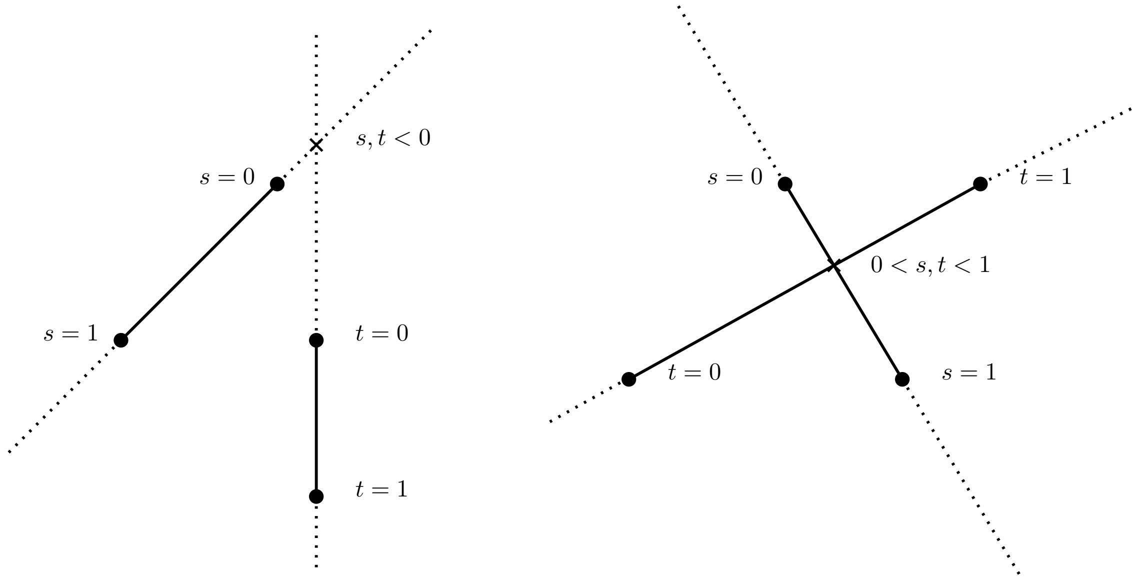


Figure 5.3: Visualization of the idea behind the formula LineIntersection

NoIntersection:
Additionally, we need to ensure is that edges cannot cross obstacle segments. For that, we can reuse the same ideas as in the planarity formula. The formulas Parallel and NoCross are nearly the same as above, only the second edge is replaced with an obstacle segment so the corresponding x and y variables are replaced with the start and end functions of the segment. We only give the higher level formula here because the implementations are just copies of the section above.

$$\text{NoIntersection} \equiv \bigwedge_{e \in E, b \in S_o} \text{Parallel}(e, b) \vee \text{NoCross}(e, b)$$

Embedding:
Finally, we need to check if the combinatorial embedding is fulfilled. For that, we simply check for every edge e on each endpoint if the next consecutive edge is the correct one according to the embedding. In order to do that, we need to check if the angle from e to $c(e)$ (we write $c(e)$ as a substitute for $c_h(e)$ or $c_t(e)$ depending on if $h(e)$ or $t(e)$ is the vertex we are checking) is smaller than the angle to the other adjacent edges on the common endpoint.

We do not give the specific formula here, but instead explain the idea behind it to ensure that the formula is expressible in ETR. For every edge $e \in E$, we ensure for both endpoints there is no other edge between e and $c(e)$, its clockwise neighbour according to the embedding. For that, we need to go through every other edge $e_2 \in E$, check if they have a common endpoint and then ensure that the angle α between e and e_2 is larger than the angle β between e and $c(e)$. For that, we use the cosinus of the angles (details how to express the cosinus in our formula in Chapter 6). As the cosinus is not linear, we need to divide the plane into two parts: the part to the right of the edge e and the part to the left of the edge e (depicted in Figure 5.4).

Now, we need to differentiate between four cases: If both edges $c(e)$ and e_2 are in the left section, we need to check if $\cos(\beta) \geq \cos(\alpha)$ as the cosinus is monotonically decreasing between 0 and 180 degrees. If $c(e)$ is in the left section and e_2 in the right one, $\beta < \alpha$, thus we accept the drawing. If it is the other way around, $\alpha < \beta$ and the embedding is not

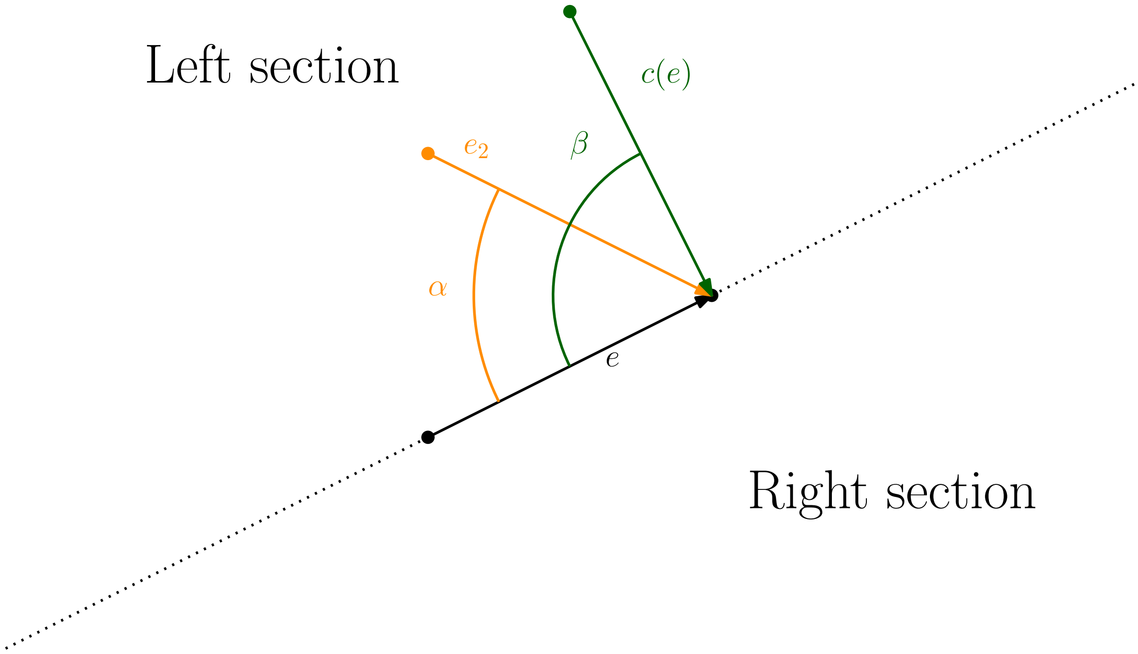


Figure 5.4: The idea behind the formula Embedding. Here, the embedding is not fulfilled as $\cos(\beta) > \cos(\alpha)$ and thus $\alpha < \beta$.

fulfilled. Finally, if both edges are in the right section, we need to check if $\cos(\beta) \leq \cos(\alpha)$ as the cosinus is monotonically increasing between 180 and 360 degrees.

5.2 ∃ℝ- Hardness

For this proof, we mainly follow the proof of Lubiw at al. [LMM18] for the problem GraphInPolygon. As outlined in Chapter 4, our main tool to use is adding additional line segments instead of the holes in the polygon in the original proof. The general idea is to reduce the known ∃ℝ-complete Planar-ETR-INV* to our problem. We do that by considering a planar incidence graph for a formula and transforming the graph into a set of segments. For that, we need different kinds of gadgets: We need a way to depict variables, we need copy gadgets to transfer the value of a variable through space, we need turn and split gadgets to replace the edges of the incidence graph and we need gadgets to implement the different kinds of formulas Planar-ETR-INV* allows: an addition gagdet, an inversion gadget and a gadget for equality to 1. We introduce the gadgets, explain why they are correct and then go in detail about how we use these gadgets to form the incidence graph.

At multiple points, we need to force edges through specific points. The way to do that is to place obstacle segments in a way that the point is not contained by either of the segments so the edge can pass that point, and to block all other possible connections between the endpoints of the edge with the line segments.

In the following figures, the black segments represent the vertex segments, their endpoints are filled as the segments are closed. The blue segments with endpoints that are not filled are the obstacle segments, their endpoints can be crossed. Vertices are represented by green crosses and edges by green lines.

Variable gadget:
The variable gadget is pretty straight forward, we can directly use the idea from the original proof, but do not need additional steps as the combination of line segments and variables

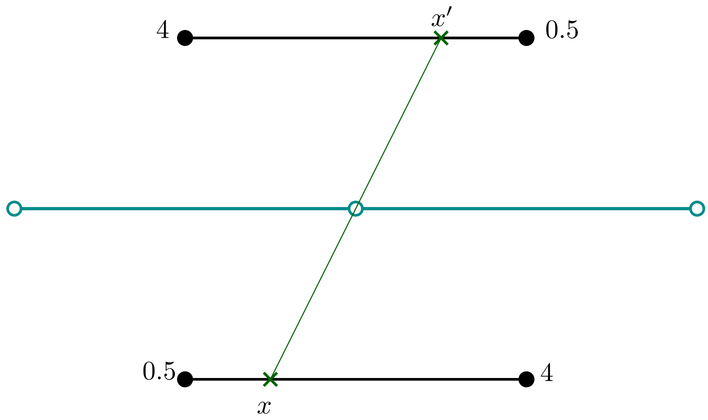


Figure 5.5: The mirror gagdet

is given by the problem. The variable gadget thus is just a line segment and the belonging vertex where the position of the vertex on the line segment indicates which value the variable holds. One endpoint of the segment holds the value 0.5, the other one the value 4. As the segment is closed, 0.5 and 4 can be represented and the segment can express exactly the values of the closed interval $[0.5, 4]$ as the problem requires.

Mirror gadget:
This gadget is not in the original proof, but seems relevant enough to get an independent section because it is the basis of a few of the other gadgets. Its idea is the mirroring of a variable gadget to be reused in other gadgets or to align the vertex segments as needed in the reduction from the incidence graph. For that, it uses two parallel variable gadgets exactly above each other and an edge between the vertices that is forced to go through the middle point of the resulting rectangle between the variable gadgets. The gadget is depicted in Figure 5.5.

Lemma 5.1. *The mirror gadget works correctly.*

Proof. The edge between the vertex segments is forced to go through the middle of the resulting rectangle, thus the distances of the vertices to the endpoints of both segments are similar, but swapped. Because of that, the position of the vertex on the top gadget is mirrored, but as the endpoints representing value 0.5 and 4 are also swapped, the segment represents the same value. □

Copy gadget:
This gagdet is meant to copy the value of a variable x, so the position of the according vertex on a line segment, onto another line segment. For that, we consecutively connect two mirror gadgets directly above each other, so the top segment of the lower mirror gadget is the base segment of the upper one. The first mirror gadget copies the value of x onto the mirrored middle variable gadget, the second one copies it onto the destination segment x'. The gadget can be seen in Figure 5.6.

Lemma 5.2. *The copy gadget works correctly.*

Proof. The copy gadget combines two mirror gadgets. As these work correctly, the value of x is mirrored on the middle segment, and then mirrored again on the top segment x'. Thus, x' has the same value and same alignement as the original segment x. □

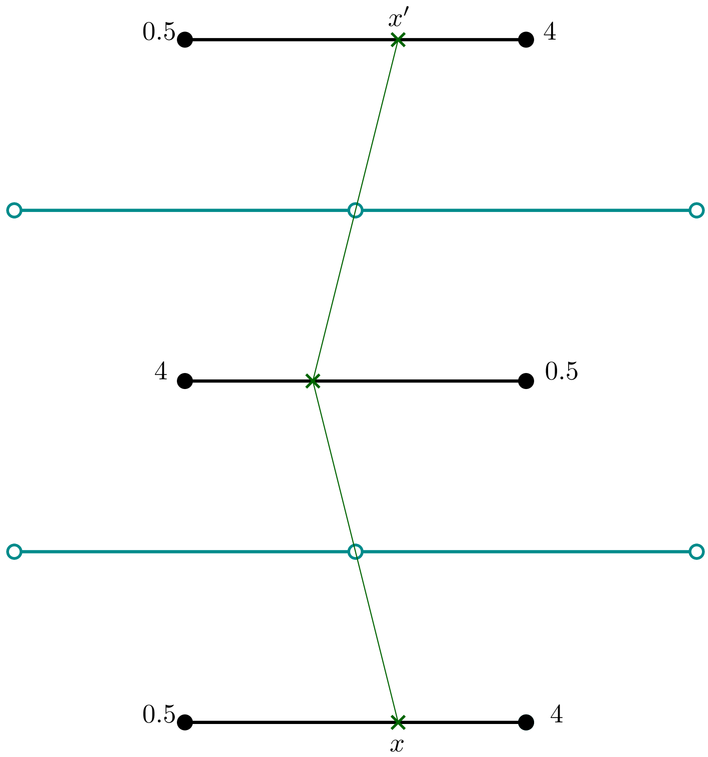


Figure 5.6: The copy gadget

Split gadget:
The split gadget is used to copy the value of a variable gadget onto two different ones to implement vertices with higher degree in the incidence graph. Here, the idea is to use two mirror gadgets, but in a skewed way so both mirror gadgets originate from the same line segment. We have one vertex segment on the bottom, and two vertex segments next to each other above it. The value of the bottom variable is copied onto the two different upper line segments (Figure 5.7).

Lemma 5.3. *The split gadget works correctly.*

Proof. Skewing the mirror gadget does not change the geometrical idea behind the gadget. The edges are now forced to go through the middle points of the resulting parallelogram, where the symmetric properties again ensure that the distance to the endpoints is mirrored in the same way as in the rectangle of the mirror gadget. □

Turn gadget:
We need the turn gadget to implement the turning points of edges in the orthogonal drawing of the incidence graph. The general idea is to use a line segment with slope $1/-1$ as a transmitter between a vertical and a horizontal variable gadget with edges from both outer vertex segments to the middle one. For that, we place our obstacle segments in a way that the point the edges can cross is on the intersection of the lines connecting the endpoints of the segments (Figure 5.8). The distances of the horizontal and the vertical segment to the diagonal transmitter segment are identical, and the transmitter segment has half the length of the normal segments.

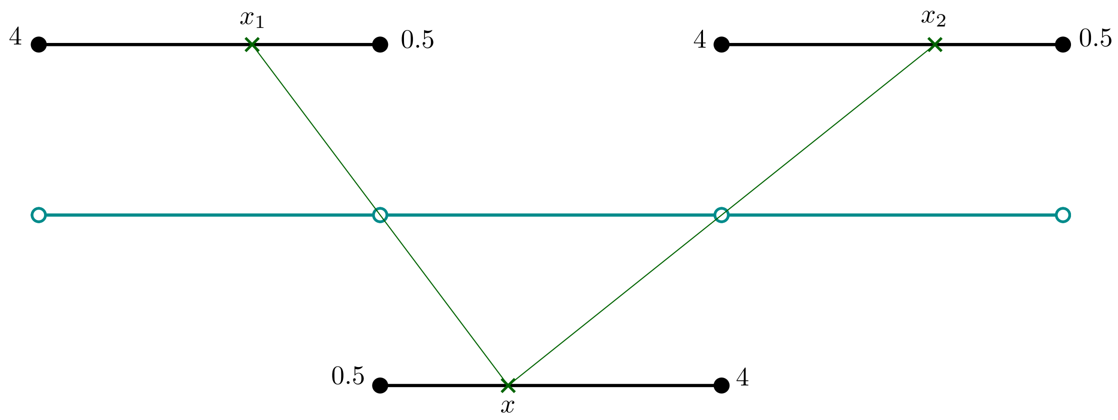


Figure 5.7: The split gadget

Lemma 5.4. *The turn gadget works correctly.*

Proof. While the relative distance to the endpoints is not the same on the diagonal segment, the gagdet works anyways because the difference can be described by a strictly monotone function (variables closer to an endpoint on the original segment are closer to the endpoint on the diagonal) and projecting the variable from the diagonal segment to the other one is the reverse operation to that function because it is a mirroring geometrical operation with the line through the diagonal segment acting as the mirror. □

Equality to 1 gadget:
This gadget represents equations of the form $x = 1$. The idea of this gadget is to build it like a mirror gadget with two vertex segments directly above each other, but add a second layer of obstacle segments to ensure that the only way the edge between the two vertices can be drawn as a straight line is on the line through the position of the variable gadget representing the value one (Figure 5.9). For that, both layers of obstacle segments only leave one point open for the edge between the vertices to cross, these points are directly under the value one of the top segment.

Lemma 5.5. *The equality to 1 gadget works correctly.*

Proof. The two points between the obstacle segments define the only line on which the edge can exist. This line only crosses the variable gagdet once, so the edge can only be drawn if the position of the lower vertex corresponds to the value one. □

Addition gadget:
This gadget is where we do not follow the approach from Miltzow but construct a different kind of gadget. It represents inequalities of the form $x + y \leq z$ or $x + y \geq z$. Here, we need that the vertex segments are pervious and also the combinatorial embedding. We also need two different constructions for $x + y \leq z$ and $x + x \geq z$. The gadget consists of two different parts: The addition part (segments x, y' and z) and the part to get the value of y onto y'. For the latter, we first need to halve the length of the variable gadget. For that, we need two additional vertex segments. The first one has normal length and is the segment that acts as the interface to the rest of the reduction. We then first halfen the length of the vertex segment. For that, we place the shorter vertex segment above the larger one in a way that the middle points of the segments are aligned. The obstacle segments now are placed in a way that the point the edge can cross is aligned with the middle points of the

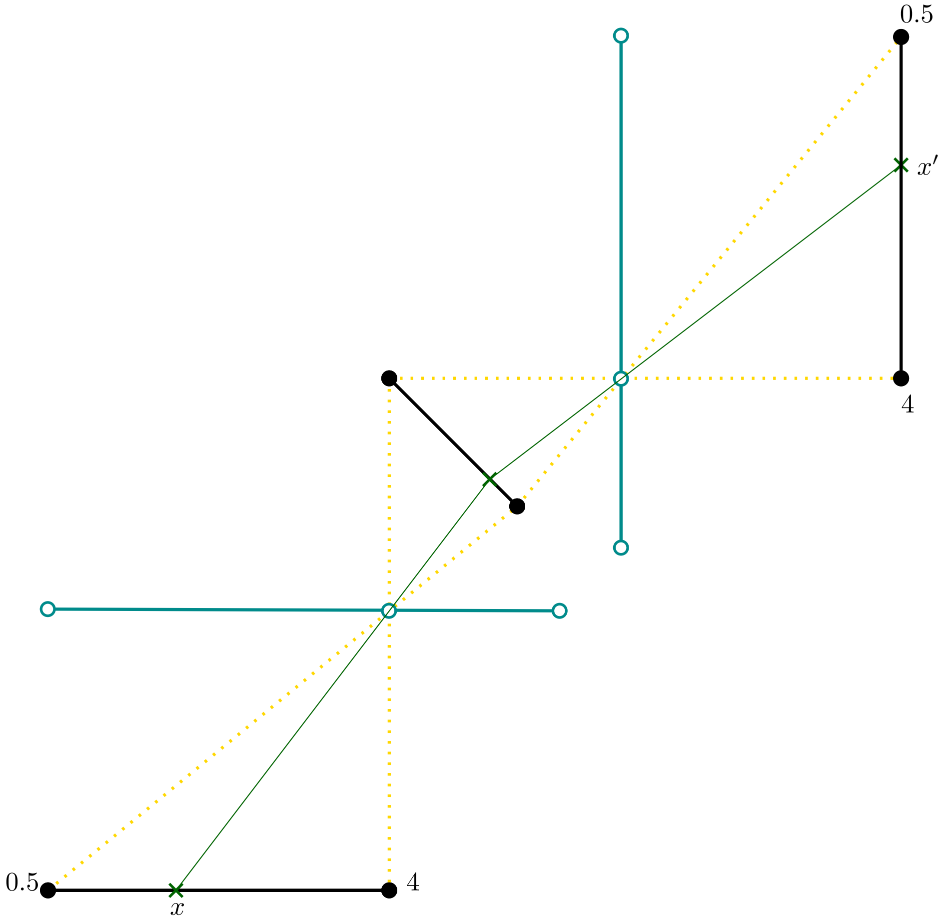


Figure 5.8: The turn gadget

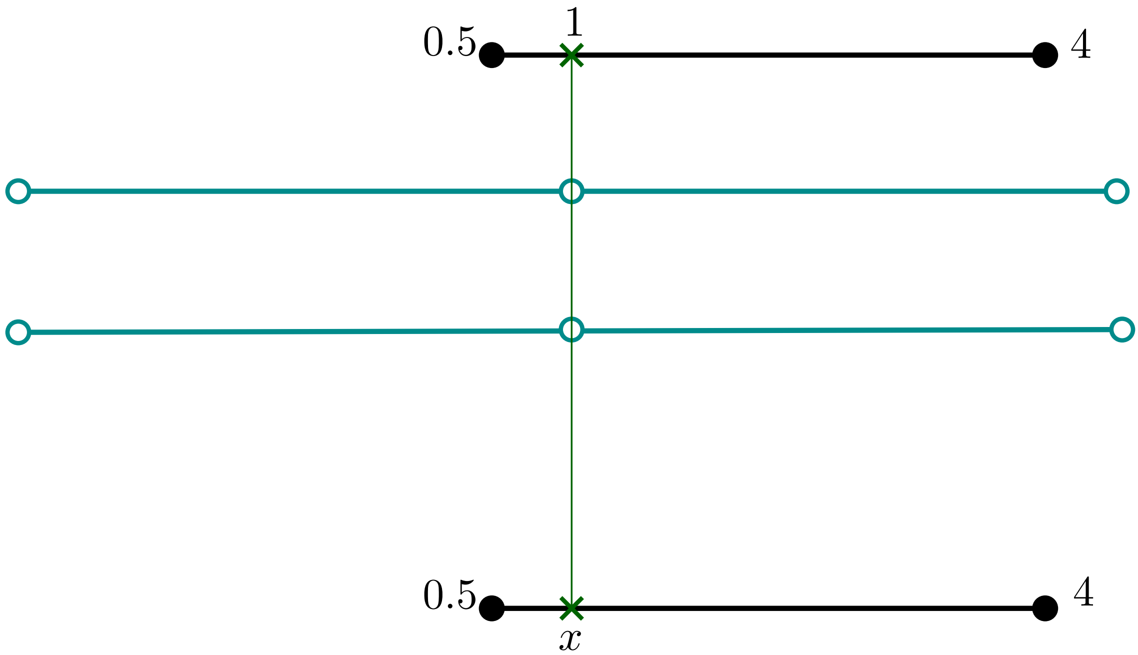


Figure 5.9: The gadget ensuring equality to one

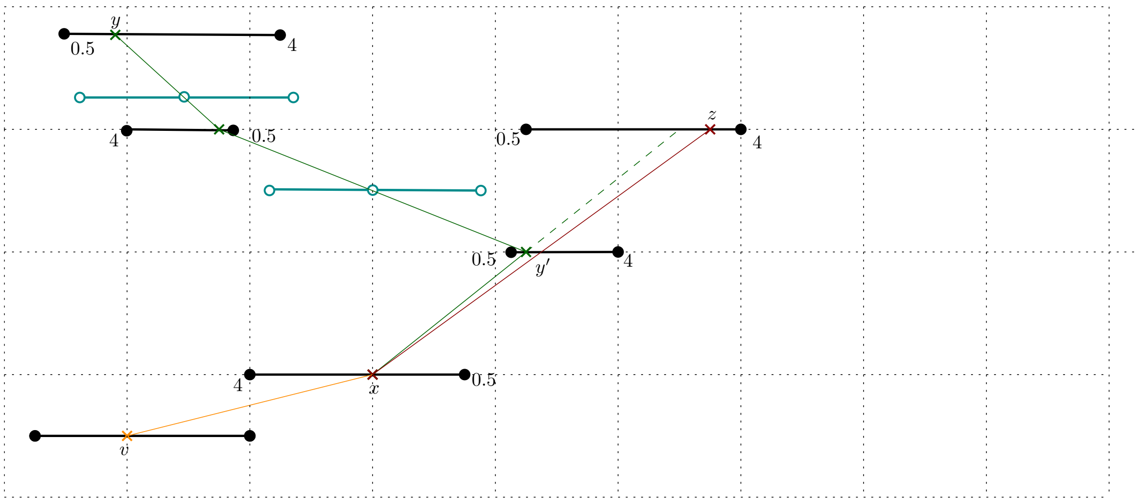


Figure 5.10: Gadget ensuring $z \geq x + y$

segments, and the distance to the smaller segment is half of the distance to the bigger one. This results in the value getting copied correctly onto the smaller segment. Finally, we copy the value of y from the middle segment onto y' with a normal copy gadget, though we need to be careful that the obstacle segments do not disturb the addition part of the gadget.

For the addition part, we have the three variable gadgets x, y' and z. The length of x and z is 7 distance units, and y' is half that long. They are parallel in 3 different layers and the distance between x and y' is the same as the distance between y' and z. The start of the y' segment is 0.75 distance units right to the end of the x gadget, and the start of the z unit is 0.25 units to the right of that (so when the gadgets would be extended to include 0, the zeros would be directly above each other). We have an edge between x and y' and an edge between x and z. For that last edge, the vertex segments have to be pervious as this edge needs to be able to cross y'. If we would extend the edge between x and y' to the z segment, it would intersect the gadget exactly on the value of $x + y$ (proof later). We use this observation to force xz to the left (or right) of the edge xy with our combinatorial embedding to ensure that z lands on a position that is $\leq x + y$ ($\geq x + y$).

That is the general structure of the gadget, for details we must distinguish between the greater than and the less than gadget:

If we want to express that $z \geq x + y$ then we need to force the edge between x and z to the right of the edge between x and y with our combinatorial embedding. We add an auxiliary vertex segment v and an edge between x and s to ensure that the embedding indeed forces xz to be to the right of xy as xz has to be on the right side of xy and on the left side of xv. The embedding for x would then be z, y', v (counter-clockwise). Due to the edge to z being to the right of xy, we need to get the value of y onto y' from the top left to not disturb the rest of the gadget (Figure 5.10).

The difference for $z \leq x + y$ is that xz is to the left of xy, so the embedding would be y', z, v (counter-clockwise). Because of that, we cannot keep the copy gadgets for y' on the top left because it would interfere with xz, instead we need to shift that to the bottom right (Figure 5.11).

Lemma 5.6. *The addition gadget is correct.*

Proof. The first thing we note is that, as stated above, if we extend the edge between x and y' to z it intersects with z directly at the value of $x + y$. That is the case because if

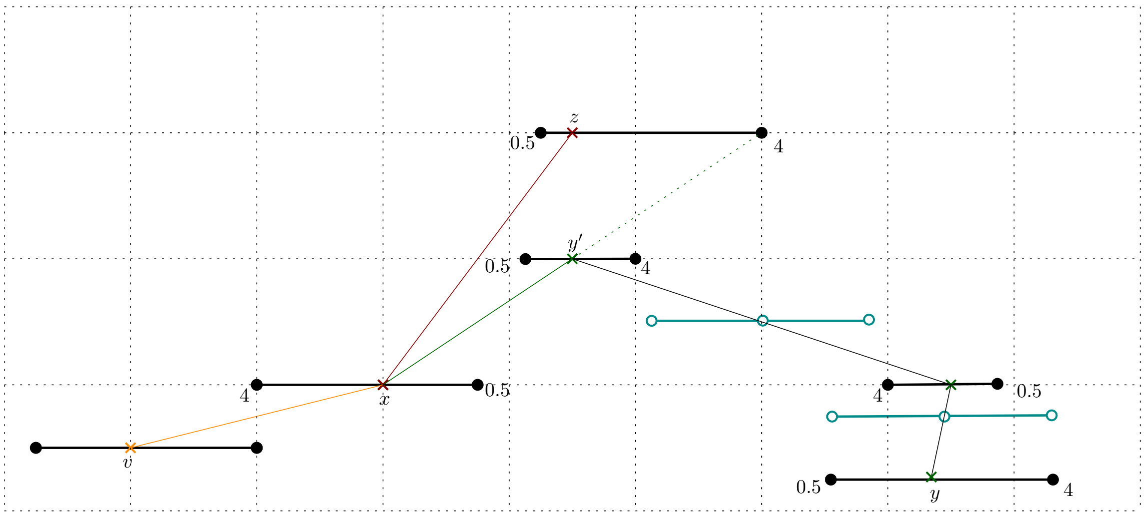


Figure 5.11: Gadget ensuring $z \leq x + y$

we fix $x = y = 0.5$, then the extension of the edge would land on 1 due to the positions of the gadgets. If we now move x to the left by a fixed length, due to the symmetry of the gadget and because y' is exactly in the middle of x and z, z moves by exactly the same amount (like a mirror gadget). If we fix x and move y, the intersection on z moves by exactly double the amount because y' is at exactly half the distance between x and z, so due to the intercept theorem z moves exactly double the distance of y'. We then force the edge between x and z to be either to the left or to the right of the edge between x and y', so the position of z has to be to the left (or right) of the position of $x + y$. That means that $z \leq x + y$ ($z \geq x + y$). We also guarantee that there is another edge from x downwards to v, so with the combinatiorial embedding we indeed force the edge between x and z to be in the right position. □

Inversion gadget:
This gadget represents the inequalities of the form $x \cdot y \leq 1$ and$x \cdot y \leq 1$. Here, we again need the distance units. We need two variable gadgets (again 7 units long) who are in a right angle with each other and that are 1.5 units apart from the point where the two lines would intersect. As PLANAR-ETR-INV* allows both $x \cdot y \leq 1$ and $x \cdot y \geq 1$, we need two different gadgets to express that. For $x \cdot y \leq 1$,we then force an edge between the variables that has to be to the right of the point p that is 1 unit from the middle point both in both dimensions, depicted in Figure 5.12. As the edge through p would exactly invert the variables, any edge to the right of p results in $x \cdot y \leq 1$. For $x \cdot y \geq 1$, we simply block the other side next to p and force the edge to be situated on the left of p.

Lemma 5.7. *The inversion gadget is correct.*

Proof. The inversion gadget is correct because the two triangles in Figure 5.12 below are similar (both have a right angle and the hypotenuses lie on the same straight line). In similar triangles the proportions of lengths of the sides are equal. That means that in our case, $\frac{x}{1} = \frac{1}{y}$, or in other words $x \cdot y = 1$ if the edge goes through p, exactly what the gadget is supposed to do. Forcing the edges to be on the right or left side of p thus results in the desired outcome of $x \cdot y \leq 1$ and $x \cdot y \geq 1$ respectively. □

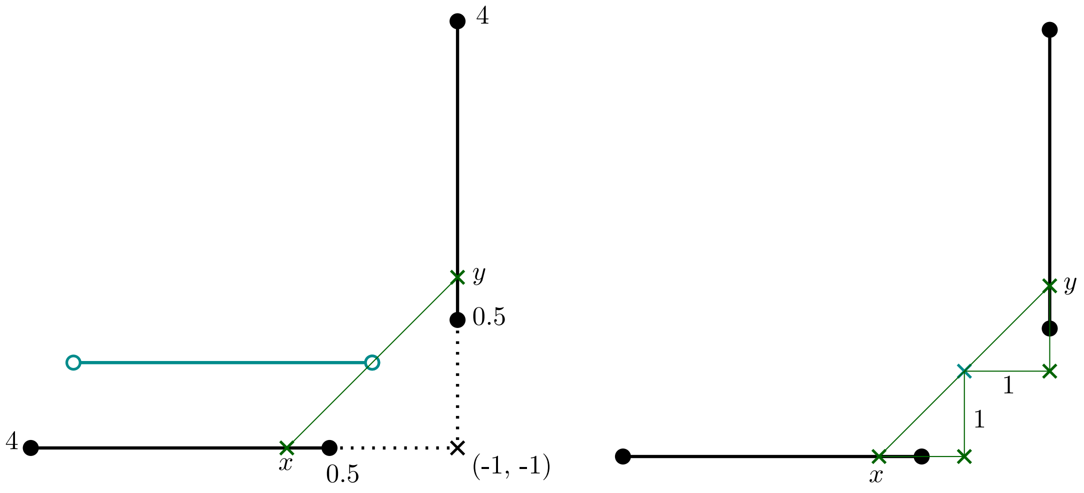


Figure 5.12: The inversion gadget

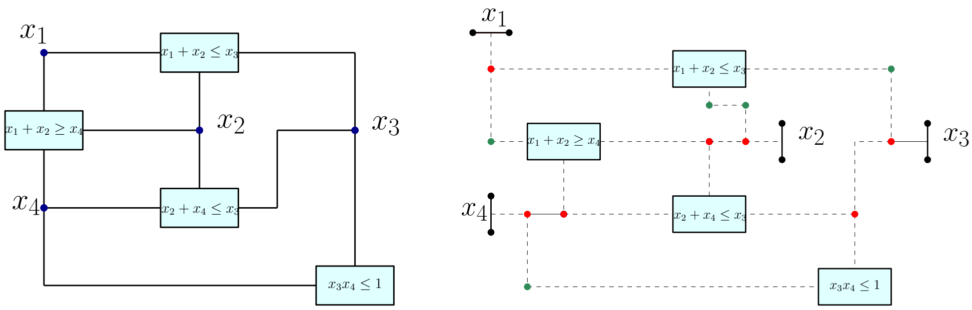


Figure 5.13: Reduction from the planar incidence graph to DrawingOnSegments. The red dots represent split gadgets, the green ones represent turn gadgets. The dashed lines are implemented with copy gadgets. (Original: [LMM18])

Incidence graph:
We now have all the tools we need to perform the reduction (depicted in Figure 5.13). The remaining question is: How do we transform the incidence graph of our Planar-ETR-INV* instance I into an instance of our problem? For that, we need an incidence graph for I. We have a vertex for each variable, a vertex for each formula and an edge between vertices if the variable is part of the formula. We then need an orthogonal drawing of the graph. Such a drawing always exists and can be computed in polynomial time as proven in [NR04]. Also, all the gadgets can be adressed properly by either rotating the gadget or using mirror and turn gadgets. The structure of the drawing also dictates the combinatorial embedding for the DrawingOnSegments instance.

Every vertex with degree greater than 1 is then replaced by multiple splitter nodes with maximum degree 3. Then every edge is replaced by a copy gadget, every 90 degree turn is replaced by a turn gadget and the variable and formula vertices are replaced by the corresponding gadgets. Every gadget works correctly as described in the lemmas above, so this reduction is both possible and also correct.

Theorem 5.8. DrawingOnSegments *is* $\exists\mathbb{R}$*-complete.*

Proof outline. We showed $\exists\mathbb{R}$-membership in Section 5.1. For $\exists\mathbb{R}$-hardness, we only give the steps of the reduction, as we proved the correctness in the earlier lemmas. We reduce from Planar-ETR-INV*, let I be an instance. The first step is to transform I into the adjusted

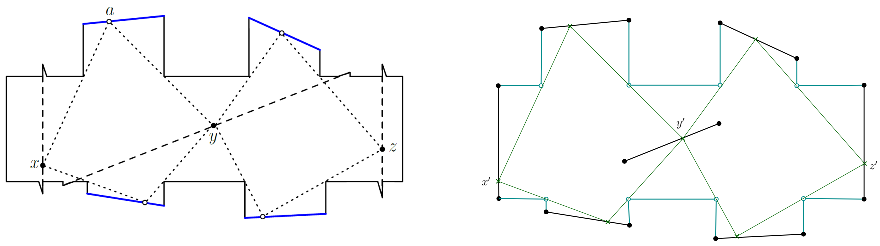


Figure 5.14: Left: The art gallery instance (Source: [LMM18]), right: the DrawingOnSegments instance

incidence graph $G(I)$ from above. Then we need an orthogonal drawing of G. Both these steps are possible in polynomial time. We transform that into a DrawingOnSegments instance as described above, which can also be done in polynomial time as all the gadgets have a constant size. □

As we mentioned earlier, it is unlikely that ∃ℝ-complete problems are in NP. One reason for that is that there are results for many ∃ℝ-complete problems which show that, even when the input consists of integers, for some problem instances the solution may require irrational coordinates and is thus not computable by a turing machine, at least not explicitly. That restricts the possibilities to find an algorithm to solve these ∃ℝ-complete problems efficiently. We now show that DrawingOnSegments also belongs in this category by giving an instance with integer coordinates for the segments that requires a solution with irrational coordinates. This instance was first found by Abrahamsen et al. for the art gallery problem [AAM18] and then modified by Lubiw et al. for GraphInPolygon [LMM18], we now adjust it for our problem:

Theorem 5.9. *There are instances of* DrawingOnSegments *where the positions of all segments can be described with integer coordinates that requires irrational coordinates for the position of at least one vertex.*

Proof outline. The original instance I of the ArtGalleryProblem and our modified instance I' of DrawingOnSegments can be seen in Figure 5.14. We follow the same approach as Lubiw et al. [LMM18] and modify I to fit our problem description. Abrahamsen et al. [AAM18] show that I can be optimally guarded by three guards which they forced on vertical guard segments. Because of the notches above and below the guard segments, an optimal solution requires guards on these segments. They show that those three guards x, y and z are enough if and only if they are placed on irrational coordinates. Note that our instance I' fulfills the same properties as I. We have replaced the guard segments for x, y and z with explicit vertex segments and the shape of the polygon with additional vertex and obstacle segments. The relevant parts of the polygon that x, y and z have to see together are replaced by vertex segments, and each of the corresponding variables to the guards that can potentially see the segment has to be adjacent to it. Thus, we also need irrational coordinates for the vertices x', y' and z'.

□

5.3 Variants

There are many changes to the problem we can discuss. We are going to focus on three questions:

- Is it possible to only allow open or only closed segments?
- Is it possible to build the gadgets without pervious segments, or maybe with only pervious segments?
- Can we reduce the amount of different slopes of the segments or the amount of different lengths of segments?

In the following paragraphs, we are one by one discussing these questions to varying results:

5.3.1 Open/Closed Segments

The first question we want to answer is if we can restrict all segments to be all open or all closed. Note that in our proof in the previous section, the segments encoding variables are all closed due to the variables lying in the closed interval $[0.5, 4]$ and the obstacle segments used to force edges through certain points are all open.

Open Segments

For now, we consider our problem, but with only open segments. That means that we have to find a different solution for our variable gadgets, because in our proof, the variables come from the closed interval $[0.5, 4]$. A possible approach to change that would be to change our version of PLANAR-ETR-INV* to PLANAR-ETR-INV, here we have variables from the open interval $(0, 5)$. We would need to adjust our gadgets for inversion and addition though to only allow equations and no inequalities, and while that works for the inversion gadgets, we would need to allow multiple edges between the same vertices to be able to force equality in the addition gadget. That would maybe lead to more problems for our formulas to prove $\exists\mathbb{R}$-membership, so we do not explore this direction further in this thesis.

Closed Segments

If we restrict all segments to be closed, we run into problems with our obstacle segments. We cannot use the constructions to force edges through certain points anymore, and can only construct corridors that lead to minimal errors in all the gadgets. A possible approach for this problem could be to determine if an alternative version of ETR-INV where each equation allows for an error ϵ is also $\exists\mathbb{R}$-complete and if that could be reducible to our problem with only closed segments. Indeed, as Delidkas et al. show [DFMS18], the $\exists\mathbb{R}$-complete variant ϵ-ETR allows deviations in the equations up to a fixed ϵ. It could be further explored if this problem also leads to an $\exists\mathbb{R}$-complete problem ϵ-ETR-INV and if our reduction could be modified in a way that the deviations in each gadgets can be bounded by an ϵ.

5.3.2 Pervious/Impenetrable Segments

Impenetrable Segments

The addition gadget is the only gadget where we use that the vertex segments are pervious, so we only need to adjust that gadget. Note that this gadget is also the only thing we need the fixed embedding for, so we can drop that restriction from our problem. Our idea for the addition gadget follows the approach from Lubiw et al. [LMM18], but it runs into a different problem as we see later. The resulting problem can be defined as follows:

Input: Graph $G = (V, E)$, set S of segments, subset $S_v \subseteq S$ of vertex segments, bijective function $f : V \to S_v$.
Problem: Is there a straight-line planar drawing of G so that each vertex v is located on the corresponding segment $f(v)$ and that the edges do not cross the segments?

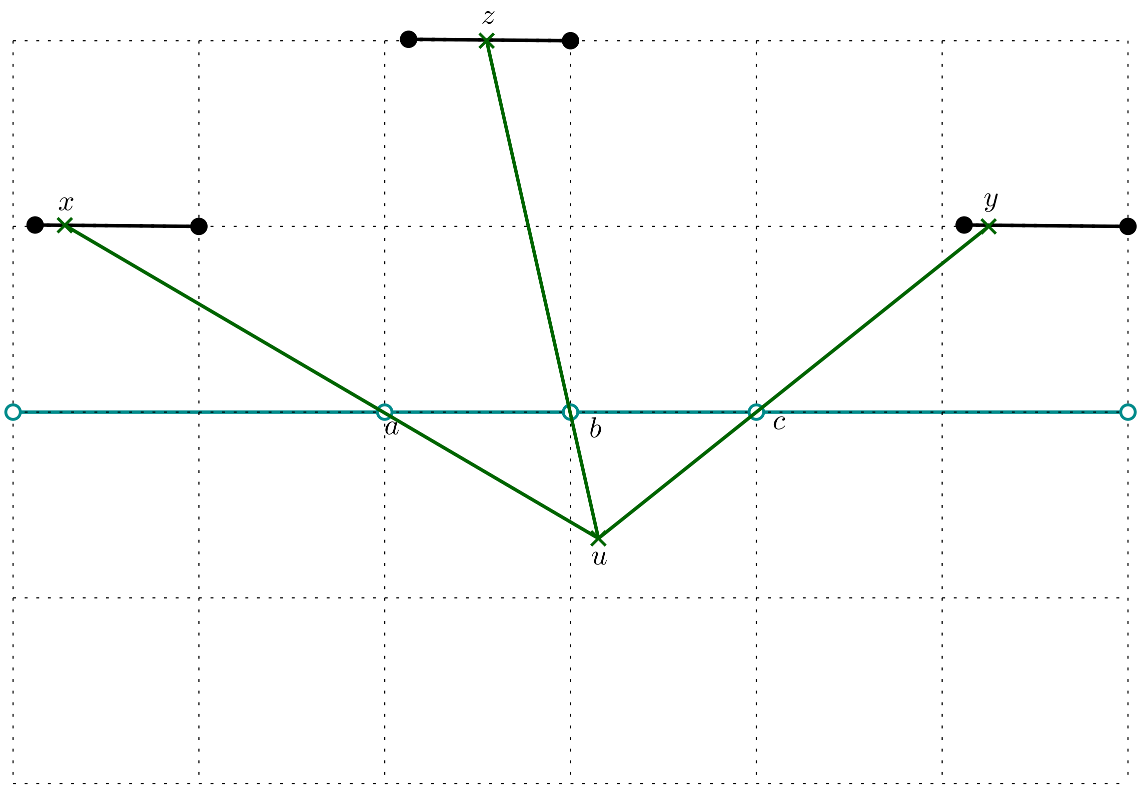


Figure 5.15: The alternative addition gadget

For this gadget, we again, we need distance units to display the relative positions of the segments. We need 3 different layers, with 8 distance units between them respectively. On the lowest layer, we need to force edges through three points (a, b and c), again 8 units apart from each other. On the layer above that, we have two variable segments for x and y, the one for x ends 8 units to the left of a and the one for y starts 9 units to the right of c. On the top layer we have another variable segment that encodes the value for z that starts 1 unit to the right of a and ends directly above b. All variable gadgets are 7 units long. We have an additional vertex u below the lowest layer that we, for now, do not locate on a segment (explanation follows). The idea is to have edges between x and u, between y and u and between u and z where the position of u dictated by x and y influences the position of z in a way that $z = x + y$. The gadget is pictured in Figure 5.15.

The problem with this idea is that we cannot use a line segment for the node u because it can lie anywhere in a quadrilateral area below the points a, b and c, so we need a quadrilateral object to capture u.

For the correctness, we need two observations: The gadget must be correct for one concrete set of variables for x and y, and it must stay correct if either x or y are changed. If both are changed, we can treat the changes for x and y sequentially. For the first part, let us set $x = y = 2$. Then both x and y are exactly the same distance from b, so their edges to u forces u to be directly under b which in turn forces z to be directly above b and thus $z = 4$, so the gadget is correct for $x = y = 2$.
For the second part, we only observe what a change to x results in, for y the same argumentation applies. Let x and y initially have some arbitrary values, and then x changes by an amount d (pictured in Figure 5.16) while y stays the same. We need to show that z moves by exactly the same amount d. Let x_1 be the initial value of x (Same argumentation if x_2 is the initial value and x becomes greater). Moving x to the left forces u to be higher on the green edge from y, which in turn forces z to move left as well, so the direction

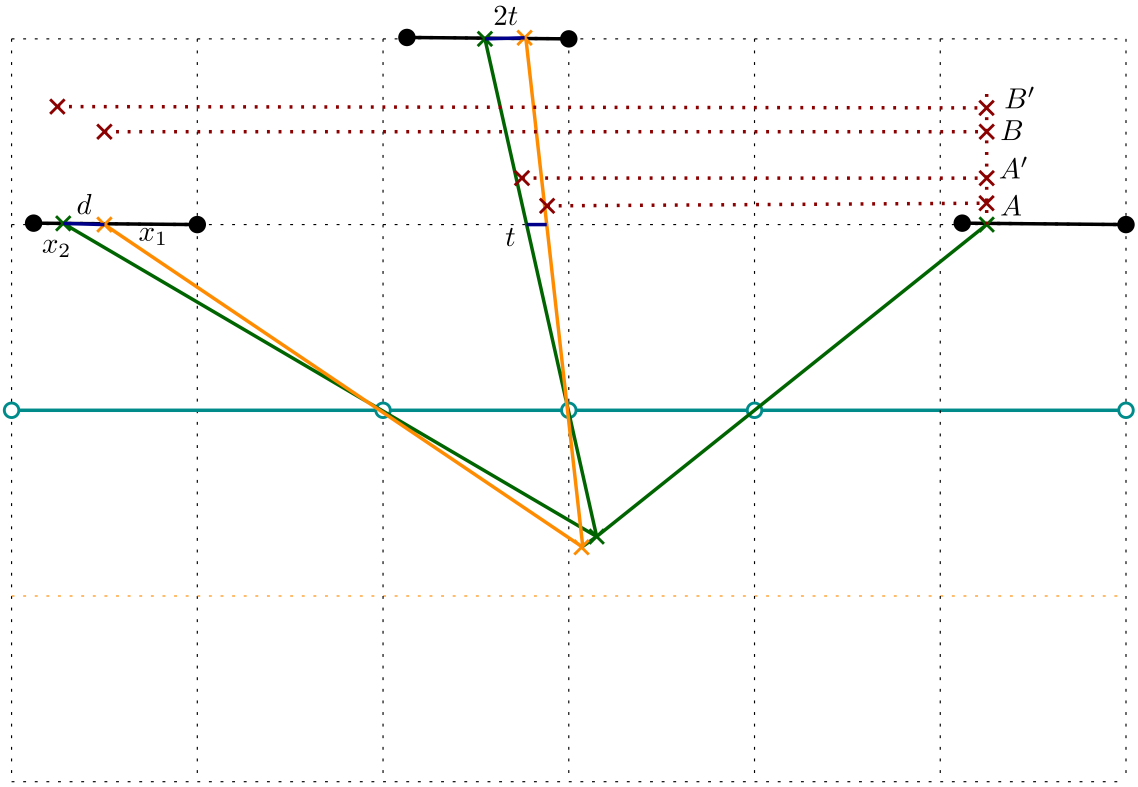


Figure 5.16: The alternative addition gadget is correct.

is right. For the distance, we look at the intersection between the edges between u and z on the level where the variable gadgets for x and y are. We note the difference by t. The distance z moves to the left is $2t$ because the variable gadget is at exactly double the distance from node b which both of the edges cross. Now the proof boils down to proving $d = 2t$. For that we define the distances A, B, A' and B': A is the distance between y and the endpoint of t in the context of x_1, B to the new endpoint of t in the context of x_2. A' is the distance between y and x_1 and B' between y and x_2. Note that the difference between A and B is t, and the difference between A' and B' is d. We know that $|A'| = 2|A|$ because the edges connecting the endpoints of A and A' with u are straight and cross the nodes a, b and c. With $|ac| = 2|bc|$, it follows that $|A'| = 2|A|$ because the relations do not change. We also know $|B'| = 2|B|$ for the same reasons. Combining this leads to $d = |A'| - |B'| = 2|A| - 2|B| = 2t$, so the gadget is correct.

To be able to contain u in an object, we need to change the problem and additionally allow vertices to be situated on quadrilaterals. The problem then changes to:

Input: Graph $G = (V, E)$, set $O = S \cup Q$ with segments S and quadrilaterals Q, bijective function $f : V \to S$.
Problem: Is there a straight-line planar drawing of G so that each vertex v is located on the corresponding object $f(v)$ and that the edges do not intersect with the objects?

This problem is $\exists\mathbb{R}$-complete because the reduction from Planar-ETR-INV* works exactly as in Section 5.2. The problem is also in $\exists\mathbb{R}$ as only the formulas OnSegment and NoIntersection change and the formula Embedding can be dropped. To handle the different objects, we would need a partition of the set of objects into the special forms and functions for each one to get information about the area the objects hold. The OnSegment formula then checks if the variable is in that area, while the NoIntersection formula checks if there are intersections between the edges and the borders of the objects. In the exact formulas for quadrilaterals, the NoIntersection formula would need to check for intersections between edges and all 4 sides of the rectangle.

Pervious Segments

If on the other hand we drop the requirement that the edges should not intersect with the segments, all of the gadgets are suddenly not possible anymore because we have no tools to force certain edges through fixed points. We do not see a way to recreate the gadgets. That is why this iteration of the problem is somewhat likely to not be $\exists\mathbb{R}$-complete.

5.3.3 Using only few Slopes/Lengths

We currently have 4 different slopes in our constructions: horizontal and vertical segments and segments with slope 1/-1 for the turn gadget. Because of the way the reduction from the incidence graph is done, we for sure need the horizontal and vertical segments. It is also not possible to drop the segments of slope 1/-1 in the turn segment, because when trying to directly convert the value of a horizontal to a vertical segment and vice versa, the ratio of the distance to the endpoints is always changed by a function f. We need to undo that function by mirroring the operation, and to end up on a segment turned by 90 degrees, we need a segment of slope 1/-1 in the middle.
When trying to unify the length of each segment, we run into similar problems with the addition gadget. Either y' or z needs to have a different length than the other, because changing the position of the node on y' causes double that change on z. However, we can restrict the lengths of all segments to either 3.5 or 7 distance units, with the shorter ones only used for the addition gadget and the turn gadget.

6. Crossing Angle Problems

In this chapter, we consider problems which restrict different kind of angles between edges of the graph, the crossing angles of edges and angles between consecutive edges on their shared endpoint, in graph drawings. Those problems could be contenders for $\exists\mathbb{R}$-completeness. We first show that they are part of $\exists\mathbb{R}$ and then discuss first attempts to show $\exists\mathbb{R}$-completeness which we predict to be the case for each of these problems.

6.1 Angular Resolution

The first problem in this category is AngularResolution, where we force the adjacent edges at a vertex to have at least a certain minimum angle α to each other to improve readability of the graph drawing:

AngularResolution:
Input: Graph $G = (V, E)$ with combinatorial embedding, angle α.
Problem: Is there a drawing of G so that the angle between each neighbouring edges of the embedding is at least α?

To show that the problem is in $\exists\mathbb{R}$, we need to give an ETR formula with size polynomial in the input size that is satisfiable exactly when there exists such a drawing of G. For that, we need a few adjustments: We assume G to be a directed graph where every undirected edge is replaced by directed edges in both directions, and again that $V = [n]$ as well as $E \subseteq \binom{[n]}{2}$. Let $h, t : E \to V$ be the head and tail functions, and $c : E \to E$ a function that returns the clockwise neighbour of e according to the embedding where the vertex connecting e and its consecutive neighbour is considered to be the head of both edges. Additionally, we need to assume that $\cos(\alpha)$ can be expressed by $q_1 \cdot \sqrt[i]{q_2}$ where $q_1, q_2 \in \mathbb{Q}, q_2 \geq 0, i \in \mathbb{N}$ because we only can express such constants in our formula and not any irrational numbers. If $q_2 \neq 1$ we need an additional variable q and the atomic formula $q \geq 0 \wedge q^i = q_2$ to encode the value of $\sqrt[i]{q_2}$ in q.

Now we can start to construct the formula ϕ. For each vertex $v \in [n]$ we need two variables (x_v, y_v) that encode the coordinates of v in the drawing. Additionally, we may need a variable l_e for each $e \in E$ that encodes the length of the edge. We need to check every angle between two neighbouring edges. If one is below the angle α, the formula has to be wrong. To check every angle, we need to look at every edge in our directed graph exactly once, because every edge is part of exactly 4 angles, two of which are then checked by the two directed edges and the other two by the counterclockwise neighbours (see Figure 6.1).

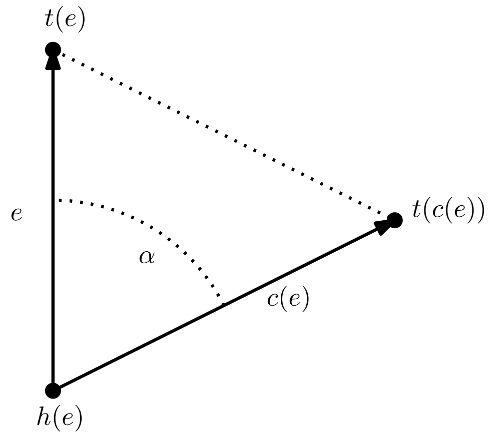


Figure 6.1: General setting for the formula

Additionally, we use the same ideas as in Section 5.1 to ensure that the drawing fulfills the embedding. Thus the general outline of our formula is the following:

$$\exists (x_1, y_1), ..., (x_n, y_n) : \bigwedge_{e \in E} \text{AtomicCheck}(e) \wedge \text{ Embedding}$$

To perform the atomic checks we need the law of cosines: In a triangle ABC with edges a, b and c, the cosinus of the angle α at A can be described by $\cos(\alpha) = (a^2 + b^2 - c^2)/(2ab)$. Using that, we can formulate the checks in the following way, although we distinguish the special case $\alpha = \pi/2$ because there we do not need the extra length variables:

Case 1: $\alpha = \pi/2$
Because of $\cos(\pi/2) = 0$, our inequality to check the condition is $(a^2 + b^2 - c^2)/(2ab) \leq 0$ which is equivalent to $a^2 + b^2 - c^2 \leq 0$, so we do not need the length variables. We now insert the correct terms for a, b and c which are the lengths of the edges e, $c(e)$ the segment between $t(e)$ and $t(c(e))$. As we only need them squared, Pythagoras allows us to replace these lengths by the squared differences of the x and y coordinates of the respective endpoints. This leads to the following formula:

$$\text{AtomicCheck}(e) \equiv ((x_{t(e)} - x_{h(e)})^2 + (y_{t(e)} - y_{h(e)})^2) + ((x_{t(c(e))} - x_{h(c(e))})^2 \\ + (y_{t(c(e))} - y_{h(c(e))})^2) - ((x_{c(e)} - x_{t(c(e))})^2 + (y_{t(e)} - y_{t(c(e))})^2) \leq 0$$

Case 2: $\alpha \neq \pi/2$
In this case, we need the length variables because we cannot eliminate $2ab$ from our formula, so the general outline changes to this:

$$\exists (x_1, y_1), ..., (x_n, y_n), l_{e_1}, ..., l_{e_m} : \bigwedge_{e \in E} (\text{Length}(e) \wedge \text{AtomicCheck}(e))$$

The atomic formula length is pretty straight forward: We use Pythagoras to get the length of the edge e encoded in l_e (visualized in Figure 6.2):

$$\text{Length}(e) \equiv ((l_e^2 = (x_{t(e)} - x_{h(e)})^2 + (y_{t(e)} - y_{h(e)})^2) \wedge l_e \geq 0)$$

For the AtomicCheck formula, we only need to change the right side of the formula: $\cos(\alpha)$ is not equal to 0 this time, so the 0 must be replaced by $\cos(\alpha) \cdot (2 \cdot l_e \cdot l_{c(e)})$:

$$\text{AtomicCheck}(e) \equiv ((x_{t(e)} - x_{h(e)})^2 + (y_{t(e)} - y_{h(e)})^2) + ((x_{t(c(e))} - x_{h(c(e))})^2 \\ + (y_{t(c(e))} - y_{h(c(e))})^2) - ((x_{t(e)} - x_{t(c(e))})^2 + (y_{t(e)} - y_{t(c(e))})^2) \leq (2 \cdot l_e \cdot l_{c(e)}) \cdot cos(\alpha))$$

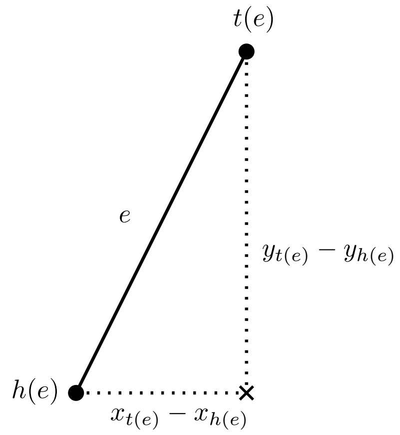


Figure 6.2: Visualization of the idea behind the formula Length

This formula is sufficient because if $\cos(\beta)$ for the angle β between e and $c(e)$ is greater than $\cos(\alpha)$, then either $\beta < \alpha$ or the remaining angle $2\pi - \beta < \alpha$ and either way there is an angle that violates the condition of the problem.

We note that, while this version of the problem admits a shorter formula, we can also show that we do not need to fix a combinatorial embedding to find a formula that solves the problem. We use mainly the same ideas as in the formula above, but replace the function c by simply checking pairs of edges, and check the angle between them if they are adjacent, that is if the head of the edges is the same. The outline of the formula is following, Length stays the same and in AtomicCheck each $c(e)$ is assumed to be replaced by e_2:

$$\begin{aligned}&\exists (x_1, y_1), ..., (x_n, y_n), l_{e_1}, ..., l_{e_m} : (\bigwedge_{e \in E} \text{Length}(e))\\ &\wedge \; (\bigwedge_{e_1, e_2, e_1 \neq e_2 \in E} h(e_1) \neq h(e_2) \;\vee\; \text{AtomicCheck}(e_1, e_2))\end{aligned}$$

6.2 Right Angle Crosing Drawings

Another way to make graph drawings easier to read for humans is to ensure that two edges only cross in a right angle to prevent edges that run almost parallel and cross. The emerging problem is the following:

RAC-Drawing
Input: Graph $G = (V, E)$.
Problem: Is there a drawing of G with straight-line edges where on each internal crossing between two edges the crossing angles are right angles?

To prove $\exists\mathbb{R}$-membership for this problem we again need a few modifications: We again assume that G is a directed graph with $V = [n]$, although this time, it is sufficient to simply orientate each edge in an arbitrary way to gain a defined head and tail for our functions h and t which we will need again.

This time, the problem requires us to look at pairs of edges and check if they cross and if they do so in a right angle. For that, we again need the variables (x_i, y_i) for the coordinates of the vertices, but we also need pairs $(s_{e_i,e_j}, t_{e_i,e_j})(i, j \in [m], i < j)$ to check for crossings between edges. Then we need to iterate over each pair of edges and check if one of three cases applies: the edges are parallel, or the edges are not parallel but do not cross, or the edges cross but do so in a right angle. The outline of the formula is the following:

$$\exists (x_1, y_1), ..., (x_n, y_n), (s_{e_1,e_2}, t_{e_1,e_2}), ..., (s_{e_{m-1},e_m}, t_{e_{m-1},e_m}) :$$
$$\bigwedge_{e_i,e_j,i<j\in E} (\text{Parallel}(e_i, e_j) \vee \ \text{NoCross}\ (e_i, e_j) \vee \ \text{RightAngleCross}(e_i, e_j))$$

Parallel:
We need to check for parallel edges because we later use a system of equations to find the crossing between the straights going to the two edges. If the edges are parallel, then there is no solution to the system of equations, so it has to be handled explicitly. We can reuse the formula from Section 5.1:

$$\text{Parallel}(e_i, e_j) \equiv (y_{t(e_i)} - y_{h(e_i)})(x_{t(e_j)} - x_{h(e_j)}) = (y_{t(e_j)} - y_{h(e_j)})(x_{t(e_i)} - x_{h(e_i)})$$

NoCross:
We again can reuse the formula from Section 5.1:

$$\text{NoCross}(e_i, e_j) \equiv \text{LineIntersection} \wedge (s_{e_i,e_j} \leq 0 \vee s_{e_i,e_j} \geq 1 \vee t_{e_i,e_j} \leq 0 \vee t_{e_i,e_j} \geq 1)$$

with the LineIntersection formula being:

$$\text{LineIntersection} \equiv s_{e_i,e_j}(x_{t(e_i)} - x_{h(e_i)}) - t_{e_i,e_j}(x_{t(e_j)} - x_{h(e_j)}) = x_{h(e_j)} - x_{h(e_i)}$$
$$\wedge \ \ s_{e_i,e_j}(y_{t(e_i)} - y_{h(e_i)}) - s_{e_i,e_j}(y_{t(e_j)} - y_{h(e_j)}) = y_{h(e_j)} - y_{h(e_i)}$$

RightAngleCross:
The final tool we need is to check if the two edges build a right angle together, so that if they cross, they fulfill the requirement of the crossing. We can use the scalar product to do that: The edges build a right angle if their scalar product is equal to 0. The remaining formula thus is the following:

$$\text{RightAngleCross}(e_i, e_j) \equiv (x_{t(e_i)} - x_{h(e_i)}) \cdot (x_{t(e_j)} - x_{h(e_j)}) + (y_{t(e_i)} - y_{h(e_i)}) \cdot (y_{t(e_j)} - y_{h(e_j)}) = 0$$

6.3 α-CrossingAngle

Now, we generalize RAC-Drawing to also allow smaller angles than 90 degrees for the crossing angle:

α-CrossingAngle
Input: Graph $G = (V, E)$, angle α.
Problem: Is there a drawing of G with straight-line edges where on each internal crossing between two edges the crossing angle is at least α?

We again make the same assumptions as with RAC-Drawing and the formula is also the same for the most part. We also need to restrict α the same way as in Section 6.2. The only thing we need to change is the last atomic formula where we need to now check for α and not for a right angle. For that we use the following formula: $\cos(\alpha) = (e_i \cdot e_j)/(|e_i| \cdot |e_j|)$. For the length of the edges we again need additional variables l_e:

$$\exists (x_1, y_1), ..., (x_n, y_n), (s_{e_1,e_2}, t_{e_1,e_2}), ..., (s_{e_{m-1},e_m}, t_{e_{m-1},e_m}), l_{e_1}, ..., l_{e_m} : \bigwedge_{e\in E} \text{Length}(e)$$
$$\wedge \bigwedge_{e_i,e_j,e_i<e_j\in E} (\text{Parallel}(e_i, e_j) \vee \text{NoCross}(e_i, e_j) \vee \alpha\text{-Cross}(e_i, e_j))$$

We reuse the formulas Length, Parallel and NoCross, only α-Cross needs to be defined. For that, we need to check if $(a \cdot b)/(|a| \cdot |b|)$ is both smaller than $\cos(\alpha)$ and greater than $-\cos(\alpha)$ because we do not know if we look at the smaller or the greater angle at the crossing and if and only if both inequalities are fulfilled, all the crossing angles are at least α. Thus, the following formula, again restructured to avoid fractions, is sufficient:

$$\begin{aligned}
&\alpha\text{-Cross}(e_i, e_j) \equiv ((x_{t(e_i)} - x_{h(e_i)}) \cdot (x_{t(e_j)} - x_{h(e_j)}) \\
&+ (y_{t(e_i)} - y_{h(e_i)}) \cdot (y_{t(e_j)} - y_{h(e_j)}) \leq cos(\alpha) \cdot l_{e_i} \cdot l_{e_j}) \\
&\vee ((x_{t(e_I)} - x_{h(e_i)}) \cdot (x_{t(e_j)} - x_{h(e_j)}) + (y_{t(e_i)} - y_{h(e_i)}) \cdot (y_{t(e_j)} - y_{h(e_j)}) \geq -cos(\alpha) \cdot l_{e_i} \cdot l_{e_j})
\end{aligned}$$

6.4 ∃ℝ-Completeness

We tried to follow the same approach as in Chapter 5 to prove ∃ℝ-completeness for each of these problems, but ran into problems along the way for all of them. It is somewhat likely that they are indeed ∃ℝ-complete as it was proven that certain special cases of the problems are NP-hard with no obvious way to show that they are also part of NP. RAC-Drawing was proven to be NP-hard by Argyriou et al. [ABS11] and AngularResolution, at least for graphs with maximum degree four, was also proven NP-hard by Formann et al. [FHH+93]. The reductions only work for variables with boolean values though, and cannot be modified to represent real values as well.

We suspect that these problems are in fact ∃ℝ-complete, without having a concrete approach for proving that. Maybe it is possible to build all the gadgets for the ETR-INV-proof, maybe it is easier to find a reduction from SimpleStretchability, but the general properties of the problems (geometrical graph drawing problems, known for being NP-hard, but not yet proven to be part of NP) are very similar to known ∃ℝ-complete problems.

7. Conclusion

In this thesis, we considered the Existential Theory of the Reals and explored the corresponding complexity class $\exists\mathbb{R}$, especially the problems that are complete in $\exists\mathbb{R}$. We showed that $\exists\mathbb{R}$ lies inbetween NP and PSPACE and found an equivalent machine modell. After that, we presented different variants of ETR, notably Feasibility, ETR-INV and the planar variants of ETR-INV in Chapter 3 as possible starting points for $\exists\mathbb{R}$-hardness reductions. We then treated the problem SimpleStretchability as another base for reductions and summarized many of those reductions that were done in the last years, notably the reductions to RECOG(SEG) and GraphInPolygon in detail. Also in Chapter 4, we presented the blueprint introduced by Miltzow et al. for reductions from ETR-INV and its planar variants and explained three reductions that follow this framework, to prove that the Art Gallery Problem, GraphInPolygon and PrescribedAreaPE are $\exists\mathbb{R}$-complete.

Our main goal was to characterize the existing reductions, find a way to replicate them easily and apply that to conduct our own $\exists\mathbb{R}$-completeness proof. As described above, we did the first part in Chapter 4. In Chapter 5, we then successfully implemented that framework for a reduction from Planar-ETR-INV* to show $\exists\mathbb{R}$-completeness of our new problem DrawingOnSegments. For that, we constructed an ETR formula for DrawingOnSegments, created the necessary gadgets for the reduction from Planar-ETR-INV* and explained the transformation from an incidence graph to the graph drawing instance. Finally, we showed $\exists\mathbb{R}$-membership for the graph drawing problems dealing with angular issues, AngularResolution, RAC-Drawing and α-CrossingAngle.

Open questions

A few questions remained open in these last two sections. While we found a few interesting facts about variants of the DrawingOnSegments problem, notably another slightly different $\exists\mathbb{R}$-complete version of the problem, there is more to explore in this setting: If we have only impenetrable segments, is there a way to design the addition gadget that does not require a quadrilateral object? If we have only closed segments, can we find an ETR-INV variant that allows us to conduct a modified $\exists\mathbb{R}$-hardness proof? And if we only have pervious segments, does the problem remain $\exists\mathbb{R}$-complete or does it maybe shift into NP?

The main open questions from our thesis are the ones from Chapter 6: Are the problems Angular Resolution, RAC-Drawing and α-CrossingAngle indeed $\exists\mathbb{R}$-complete? They are NP-hard, at least for special cases, with no obvious way to prove NP-membership, and we showed that they belong in $\exists\mathbb{R}$. Because of that, we believe that it is likely that they could be shown to be $\exists\mathbb{R}$-complete in the future and gave some first approaches how such a proof could be conducted. While our attempts to create all the gadgets necessary for an ETR-INV based $\exists\mathbb{R}$-hardness proof have failed, maybe there are ways to do it successfully. It could also be easier to base the reductions on SimpleStretchability, or other $\exists\mathbb{R}$-complete problems that are more similar to these angular problems.

Bibliography

[AAM18] Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 65–73, 2018.

[ABB+20] Oswin Aichholzer, Manuel Borrazzo, Prosenjit Bose, Jean Cardinal, Fabrizio Frati, Pat Morin, and Birgit Vogtenhuber. Drawing graphs as spanners. *arXiv preprint arXiv:2002.05580*, 2020.

[ABS11] Evmorfia N Argyriou, Michael A Bekos, and Antonios Symvonis. The straight-line RAC drawing problem is NP-hard. In *International Conference on Current Trends in Theory and Practice of Computer Science*, pages 74–85. Springer, 2011.

[AM19] Mikkel Abrahamsen and Tillmann Miltzow. Dynamic Toolbox for ETRINV. *arXiv preprint arXiv:1912.08674*, 2019.

[AMS20] Mikkel Abrahamsen, Tillmann Miltzow, and Nadja Seiferth. Framework for $\exists\mathbb{R}$-Completeness of Two-Dimensional Packing Problems. *arXiv preprint arXiv:2004.07558*, 2020.

[BC06] Peter Bürgisser and Felipe Cucker. Counting complexity classes for numeric computations II: Algebraic and semialgebraic sets. *Journal of Complexity*, 22(2):147–191, 2006.

[Bie91] Daniel Bienstock. Some provably hard crossing number problems. *Discrete & Computational Geometry*, 6(3):443–459, 1991.

[BM16] Vittorio Bilò and Marios Mavronicolas. A catalog of EXISTS-R-complete decision problems about Nash equilibria in multi-player games. In *33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[Bro03] Christopher W Brown. QEPCAD B: a program for computing with semi-algebraic sets using CADs. *ACM SIGSAM Bulletin*, 37(4):97–108, 2003.

[BSS+89] Lenore Blum, Mike Shub, Steve Smale, et al. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bulletin (New Series) of the American Mathematical Society*, 21(1):1–46, 1989.

[Can88] John Canny. Some algebraic and geometric computations in PSPACE. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 460–467, 1988.

[Car15] Jean Cardinal. Computational geometry column 62. *ACM SIGACT News*, 46(4):69–78, 2015.

[CH17] Jean Cardinal and Udo Hoffmann. Recognition and complexity of point visibility graphs. *Discrete & Computational Geometry*, 57(1):164–178, 2017.

[CK13] Jean Cardinal and Vincent Kusters. The complexity of simultaneous geometric graph embedding. *arXiv preprint arXiv:1302.7127*, 2013.

[Col75] George E Collins. Quantifier elimination for real closed fields by cylindrical algebraic decompostion. In *Automata Theory and Formal Languages 2nd GI Conference Kaiserslautern, May 20–23, 1975*, pages 134–183. Springer, 1975.

[Coo71] Stephen A Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158, 1971.

[DFMS18] Argyrios Deligkas, John Fearnley, Themistoklis Melissourgos, and Paul G Spirakis. Approximating the existential theory of the reals. In *International Conference on Web and Internet Economics*, pages 126–139. Springer, 2018.

[DGC99] Ernest Davis, Nicholas Mark Gotts, and Anthony G Cohn. Constraint networks of topological relations and convexity. *Constraints*, 4(3):241–280, 1999.

[DKMR18] Michael Gene Dobbins, Linda Kleist, Tillmann Miltzow, and Paweł Rzążewski. $\forall\exists\mathbb{R}$-Completeness and Area-Universality. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 164–175. Springer, 2018.

[Eri19] Jeff Erickson. Optimal Curve Straightening is $\exists\mathbb{R}$-Complete. *arXiv preprint arXiv:1908.09400*, 2019.

[FHH+93] Michael Formann, Torben Hagerup, James Haralambides, Michael Kaufmann, Frank Thomson Leighton, Antonios Symvonis, Emo Welzl, and G Woeginger. Drawing graphs in the plane with high resolution. *SIAM Journal on Computing*, 22(5):1035–1052, 1993.

[Gra07] Erich Graedel. Finite model theory and descriptive complexity. In *Finite model theory and its applications*, pages 125–230. Springer, 2007.

[GVJ88] D Yu Grigor'ev and Nikolaj N Vorobjov Jr. Solving systems of polynomial inequalities in subexponential time. *Journal of symbolic computation*, 5(1-2):37–64, 1988.

[H+91] Hoon Hong et al. *Comparison of several decision algorithms for the existential theory of the reals.* Citeseer, 1991.

[Hof17] Udo Hoffmann. On the Complexity of the Planar Slope Number Problem. *J. Graph Algorithms Appl.*, 21(2):183–193, 2017.

[HRS93] Joos Heintz, M-F Roy, and Pablo Solernó. On the theoretical and practical complexity of the existential theory of reals. *The Computer Journal*, 36(5):427–431, 1993.

[Kyn11] Jan Kynčl. Simple realizability of complete abstract topological graphs in p. *Discrete & Computational Geometry*, 45(3):383–399, 2011.

[LMM18] Anna Lubiw, Tillmann Miltzow, and Debajyoti Mondal. The complexity of drawing a graph in a polygonal region. In *International Symposium on Graph Drawing and Network Visualization*, pages 387–401. Springer, 2018.

[Mat14] Jiri Matousek. Intersection graphs of segments and $\exists\mathbb{R}$. *arXiv preprint arXiv:1406.2636*, 2014.

[MM13] Colin McDiarmid and Tobias Müller. Integer realizations of disk and segment graphs. *Journal of Combinatorial Theory, Series B*, 103(1):114–143, 2013.

[Mnë88] Nikolai E Mnëv. The universality theorems on the classification problem of configuration varieties and convex polytopes varieties. In *Topology and geometry—Rohlin seminar*, pages 527–543. Springer, 1988.

[NR04] Takao Nishizeki and Md Saidur Rahman. *Planar graph drawing*, volume 12. World Scientific Publishing Company, 2004.

[Ren88] James Renegar. A faster PSPACE algorithm for deciding the existential theory of the reals. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.

[Sch09] Marcus Schaefer. Complexity of some geometric and topological problems. In *International Symposium on Graph Drawing*, pages 334–344. Springer, 2009.

[Sch13] Marcus Schaefer. Realizability of graphs and linkages. In *Thirty Essays on Geometric Graph Theory*, pages 461–482. Springer, 2013.

[Sch14] Marcus Schaefer. Picking planar edges; or, drawing a graph with a planar subgraph. In *International Symposium on Graph Drawing*, pages 13–24. Springer, 2014.

[Sho] Peter W Shor. Stretchability of pseudolines is NP-hard, Applied geometry and discrete mathematics, 1991. *MR1116375 (92g: 05065)*, pages 531–554.

[SŠ17] Marcus Schaefer and Daniel Štefankovič. Fixed points, Nash equilibria, and the existential theory of the reals. *Theory of Computing Systems*, 60(2):172–193, 2017.

[Tar98] Alfred Tarski. A decision method for elementary algebra and geometry. In *Quantifier elimination and cylindrical algebraic decomposition*, pages 24–84. Springer, 1998.