COMPLEXITY ISSUES IN DYNAMIC GEOMETRY

JÜRGEN RICHTER-GEBERT AND ULRICH H. KORTENKAMP

ABSTRACT. This article deals with the intrinsic complexity of tracing and reachability questions in the context of elementary geometric constructions. We consider constructions from elementary geometry as *dynamic entities*: while the free points of a construction perform a continuous motion the dependent points should move consistently and continuously. We focus on constructions that are entirely built up from *join*, *meet* and *angular bisector* operations. In particular the last operation introduces an intrinsic ambiguity: Two intersecting lines have *two* different angular bisectors. Under the requirement of continuity it is a fundamental algorithmic problem to resolve this ambiguity properly during motions of the free elements.

After formalizing this intuitive setup we prove the following main results of this article:

- It is NP-hard to trace the dependent elements in such a construction.
- It is NP-hard to decide whether two instances of the same construction lie in the same component of the configuration space.
- The last problem becomes PSPACE-hard if we allow one additional sidedness test which has to be satisfied during the entire motion.

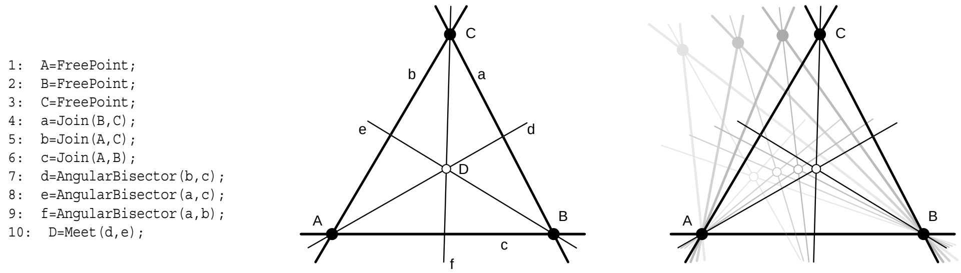
On the one hand the results have practical relevance for the implementations of Dynamic Geometry Systems. On the other hand the results can be interpreted as statements concerning the intrinsic complexity of analytic continuation.

1. INTRODUCTION

1.1. **What is Dynamic Geometry.** Imagine any construction of elementary geometry – for instance, a ruler and compass construction of the midpoint of two points *A* and *B*. It consists of certain *free* elements (the points *A* and *B*) and certain *dependent* elements whose positions are determined by the positions of the free elements. Each specific drawing of such a construction is a snapshot that belongs to the whole continuum of all possible drawings for all possible locations of the free elements. If we move the free elements we can walk continuously from one *instance* (i.e. snapshot) of the construction to another one. During such a walk a continuous motion of the free elements should result in a continuous movement of the dependent elements.

This article deals with those effects and problems that genuinely arise from such a dynamic and continuous setup of geometry. The research that led to the results presented in this article was motivated by the desire (and the actual work) of implementing a software package for doing Dynamic Geometry on a computer [22, 23]. With such a program one should be able to do constructions of elementary geometry with a few mouse clicks, and after this pick the free elements with the mouse – drag them around – while the whole construction follows accordingly. The unsuspicious looking requirement of *continuity of dependent elements* turned out to be fundamentally hard to fulfill. In fact, one has to rely on notions of complex function theory and Riemann surfaces to get a mathematically sound treatment of these effects [11, 12]. While this is no problem in theory, we prove here that

Keywords: Dynamic Geometry, analytic continuations, NP, PSPACE, complexity, reachability, ruler and compass, linkages, warehouseman's problem



```
A=FreePoint;
B=FreePoint;
C=FreePoint;
a=Join(B,C);
b=Join(A,C);
c=Join(A,B);
d=AngularBisector(b,c);
e=AngularBisector(a,c);
f=AngularBisector(a,b);
 D=Meet(d,e);
```

Construction sequence static picture idea of the dynamic picture

Fig. 1: Dynamic behavior of the angular bisector theorem.

from a complexity theoretic point of view most algorithmic questions related to that context are provably intractable (unless P=NP, of course). The complexity classes that arise here range from NP-hard problems via PSPACE-hard problems up to even undecidable problems. In particular we prove that …

- … it is NP-hard to calculate the positions of the dependent elements after a specific move of a free element (Sec. 5),
- … in general, it is PSPACE-hard to decide whether two instances of the same construction can be continuously deformed into each other if all free and dependent elements must have real coordinates (Sec. 6),
- … this reachability problem is still NP-hard if only *join*, *meet*, and *angular bisector* operations occur (Sec. 4),
- … it is undecidable whether two instances of a construction involving "wheels" – devices that transfer angles to distances – can be continuously deformed into each other by moving the base elements (Sec. 7).

Although the results of this article arose from the study of configuration spaces of elementary geometric constructions they are naturally related to many other setups in the area of geometry. Among those are the study of configuration spaces of mechanical linkages [6, 9, 10], realization spaces of oriented matroids [16, 4, 20, 25] and polytopes [21], and the warehouseman's problem [7, 24]. The results of this article are partially generalizations and strengthenings of known complexity results in these areas. Besides the context of Dynamic Geometry our results are relevant for all areas where geometric objects are moved around under certain geometric constraints, like robotics, parametric CAD [5], virtual reality, or computational kinematics. Our results imply that many problems of these areas are computationally difficult (like the *persistent naming problem* of parametric CAD [5] or the *navigation problem* of computational kinematics). Also one can interpret the results of this paper as statements on the complexity of analytic continuation (all coordinate functions in our setup turn out to be analytic). In particular this gives intrinsic complexity bounds on homotopy methods for solving polynomial equations as they were discussed in [26, 27, 28, 29, 30]. This article is complemented by [11, 12] were we give conceptual approaches to handle a dynamic setup of geometry at all.

1.2. **Constructions, Forbidden Situations and Ambiguities.** In a typical setup for this article we will study construction sequences in which each single construction step is of very elementary nature like taking the *join of two points*, the *meet of two lines*, the *angular bisector of two lines*, or the *intersection of a line and a circle*, etc. A construction sequence

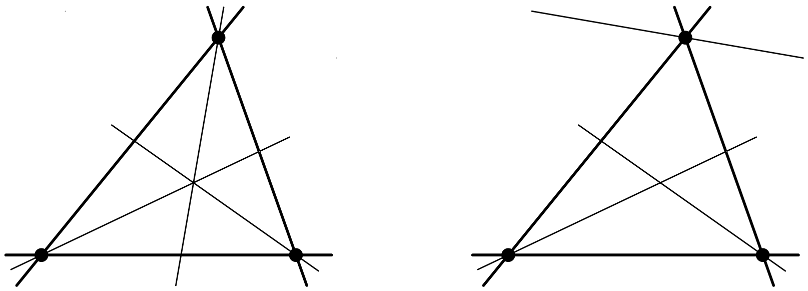


Fig. 2: Two instances of the angular bisectors of a triangle.

starts with some free points and generates new elements by performing elementary operations on already existing elements one at a time. It may happen that an operation cannot be carried out (for instance, if one wants to construct the join of two identical points, the meet of two identical or parallel lines, or the intersection of a line and a circle that do not meet). In order to avoid such situations let us assume that the input points are in suitable positions such that each step of the construction sequence can be done. In that case we will call the input point position *admissible*, otherwise we call it *forbidden*.

The *join* and *meet* operations are *deterministic* construction steps in the sense that for each admissible input there exists exactly one corresponding output element (for instance two distinct, non-parallel lines have exactly one point of intersection). Construction sequences that exclusively involve join and meet operations are easy to handle: If for a certain position of the free elements each construction step is admissible then the positions of the dependent elements are uniquely determined.

The situation is substantially different for operations like *intersection of a circle and a line,* or *angular bisector of two lines*. For these operations one has a binary choice of what the output of an operation should be (two lines have two angular bisectors, a line and a circle have in general two points of intersection). For a construction involving such operations the positions of the dependent elements are no longer uniquely determined by the positions of the free elements. This kind of *non-determinism* will be captured by the concept of a *geometric straight line program*, which is formalized in Sec. 2 (see also [11]).

The intrinsic ambiguities of these operations together with continuity requirements are the fundamental sources that make the algorithmic problems studied in this article difficult. These intrinsic ambiguities even touch the very heart of the notion of *"What is a geometric theorem?"* Consider the theorem stating that *the angular bisectors of the sides of a triangle meet in a point*. Due to the intrinsic ambiguity of the angular bisector operation this sentence stated as such is not true. Consider the drawing in Fig. 2. It shows two valid instances of the construction: Take three points – form the three joins of any pair of them – draw the three angular bisectors of any pair of lines. In the left drawing the chosen angular bisectors meet, in the right drawing they do not.

Having these ambiguities in mind, in the context of Dynamic Geometry two natural questions arise:

- **Reachability problem:** Is it possible to move the free points such that a first instance is smoothly deformed into a specific second one?
- **Tracing problem:** How can a Dynamic Geometry program decide after a motion of the free elements what instance to draw for the new position?

After a suitable formalization, we will show that the reachability problem is in general PSPACE-hard. It is still NP-hard if one restricts oneself to constructions that only use *join*, *meet*, and *angular bisector* operations. The tracing problem turns out to be (at least) NP-hard.

1.3. **Restricting the Operations.** We try to formulate our statements as strongly as possible and restrict the allowed elementary operations to a minimum. The only operations we will use are *join*, *meet*, *angular bisectors* and *intersection of circle and line*. Furthermore, we assume that initially four fixed *constant* base points $(0,0)$, $(1,0)$, $(0,1)$ and $(1,1)$ are given. In fact, we try to use the *intersection of circle with line* operation as sparsely as possible. The reason for this is that the possible non-existence of such an intersection includes the possibility to encode sidedness conditions and "cutting holes" in the admissible range of input parameters. By explicitly excluding operations like *intersection of circle and line* we substantially strengthen our results. In fact the NP-hardness results for the tracing and reachability problems can be exclusively stated in terms of angular bisectors, join and meet. The PSPACE-hardness results need just one single intersection-of-circle-and-line operation. We also try to introduce as few free points as possible into or constructions. This complements many other related complexity results since there usually many free variables are needed. The following table summarizes the complexity results covered in this article:

Problem	**Complexity**	#free points	#angular bisectors	# int. circle/lines	# wheels
Tracing	NP-hard	1	many	—	—
Reachability	NP-hard	many	3	—	—
Reachability	NP-hard	1	many	—	—
Reachability	PSPACE-hard	1	many	1	—
Reachability	PSPACE-hard	many	—	1	—
Reachability	Undecidable	1	2	—	11

In order to exclude unnecessary technicalities arising from special cases we also assume that the plane is extended by elements at infinity to the usual projective plane.

The results that use exclusively angular bisectors (and join and meet) are in a sense generalizations of corresponding results in other setups in the following sense. While with mechanical linkages or with ruler and compass constructions it is easily possible to construct angular bisectors, the converse is impossible. A complexity theoretic lower bound for a setup that uses only angular bisectors is therefore a stronger result than a corresponding one for linkages or ruler and compass constructions.

1.4. **Related Results.** There are other related areas of geometry where similar complexity results arise. In this section we want to briefly discuss the relations – similarities and differences – to these results.

1.4.1. *Oriented Matroids and Polytopes.* Research over the last few decades showed that for oriented matroids and polytopes so called *universality theorems* can be proved. These theorems show that the corresponding realization spaces can essentially be (stably equivalent to) any solution space of a system of (finitely many) polynomial equations and inequalities [16, 4, 20, 21, 25]. These results are usually derived by a direct translation procedure, which starts from a system of polynomials and ends up with a configuration of the desired category (an oriented matroid or a polytope). In the realization space of the oriented matroid (or polytope) the original variables of the algebraic equations can be rediscovered from the coordinates of certain points. The constructions of universality theorems deeply rely on the generation of large loops that feedback the result of an evaluation of a polynomial to a initially chosen constant (for instance a point whose coordinates represent the "1".) Deciding whether the realization space of an oriented matroid or polytope is empty or not turns out to be NP-hard. Deciding whether two realizations of an oriented matroid (or polytope) are in the same connected component of the realization space turns out to be PSPACE-hard.

In Sec. 4 although we derive a similar result for elementary geometric constructions where we rely on very different effects. The complex behavior is generated by a strictly forward oriented construction without any feedback of information. Orientation information cannot and is not included in any way

IFor the PSPACE-hardness result in Sec. 6. the significant difference to the constructions for oriented matroids or polytopes is that we use only *one* free point instead of *many* free points.

1.4.2. *Mechanical Linkages.* A similar comparison holds for mechanical linkages, for which universality theorems are known [6, 9, 10] that prove that arbitrary primary semi-algebraic sets can show up as components of configuration spaces. The corresponding reachability problem ("Do two instances of a linkage lie in the same component of the configuration space?") is also PSPACE-hard.

The results there are obtained by making use of construction loops and inequality relations. The inequality relations arise naturally in that context, since the bars of a linkage are only of finite length. In our setup only weaker construction primitives are allowed. Furthermore also the linkage results rely on the introduction of *many* free elements, in contrast to our results.

1.4.3. *Warehouseman's Problem.* Moving an object with a non-restricted number of degrees of freedom through a world of geometric obstacles leads to another PSPACE-hard reachability problem [7, 24]. Again, the use of inequality relations is already inherent in the statement of the problem, and many free elements are needed.

1.5. **Acknowledgements.** We want to thank Alexander Below, Vanessa Krummeck and Jesus de Loera for careful proofreading and many valuable discussions. We want to thank Maurice Rojas for drawing our attention to Plaisted's Theorem that served as a paradigm for the proof in Sec. 5 and simplified our original construction. We also thank Yuri Matiyasevich who supplied us with the proper reference for the best known bound for the number of variables that is needed to prove the undecidability of Hiberts 10th problem over the integers (see Sec. 7).

2. Geometric Constructions

2.1. **Geometric Straight Line Programs.** We now start to formalize the concept of a geometric construction. We take special care to have a setup that allows the results of an operation to be non-existent or ambiguous. For this we first define the notion of a *relational instruction set*. Here, instead of giving an algorithm or formula for the operation, only a relation is specified that enables us to check the validity of a certain input and output pair. A slightly more general approach can be found in [11].

Definition 2.1. A *relational instruction set (RIS)* is a pair (O,Ω) of *objects* O and *primitive operations* Ω with the following properties: $O = (O_1,\dots,O_k)$ is a family of sets O_i. These sets partition the objects into classes of the same *type*. The primitive operations $\Omega = (\omega_1,\dots,\omega_l)$ are relations

$$\omega_i \subset (O_{x^i_1} \times \cdots \times O_{x^i_{s_i}}) \times O_{x^i_{s_{i+1}}}$$

with *input size (arity)* $\mathrm{ar}(\omega_i) = s_i$ and *type* $\mathrm{type}(\omega_i) := O_{x^i_{s_{i+1}}}$. An element $(o_1,\dots,o_{\mathrm{ar}(\omega)}) \in O_{x^i_1} \times \cdots \times O_{x^i_{s_i}}$ is called an *input* and an element $o \in O_{x_{s_i+1}}$ is called an *output* of ω_i.

Remark 2.2. For the relation $(o_1,\dots,o_{\mathrm{ar}(\omega)},o) \in \omega_i$ we will also use the more intuitive notation

$$o \leftarrow \omega_i(o_1,\dots,o_{\mathrm{ar}(\omega)}).$$

This notion may be considered as a non-deterministic assignment operation. It assigns to an input $(o_1,\dots,o_{\mathrm{ar}(\omega)})$ one of the potential outputs o of ω. However, one should have in mind that this notion still represents a *relation* that can be *true* or *false*. It is true if the input is admissible for the operation and if the output is one of the proper evaluations of ω on this input.

In our geometric setup the different classes of objects will correspond to points, lines, circles, etc. Each primitive operation will represent a certain type of geometric primitive construction like join, meet, angular bisectors, etc. In addition, we will allow special operations to create free points which will play the role of the "input" of our constructions. Observe that relational instruction sets are general enough to describe not only geometric, but also arithmetic operations (see [11]).

We now describe the specific objects and operations used in this article. Although we will make use of Euclidean operations, we will describe the purely incidence geometric part for points and lines in terms of projective geometry. This will exclude unnecessary special cases and helps in defining the right concept of continuity later on. We embed everything in the real projective plane $\mathbb{RP}^2$. In the usual way we can represent points and lines (in homogeneous coordinates) by vectors in $\mathbb{R}^3 \setminus \{(0,0,0)\}$. Vectors that only differ by a scalar multiple are identified and represent the same point (or line). A point (x,y,z) is on a line (a,b,c) if and only if $ax+by+cz=0$. Meet and join can then be simply expressed as cross-products of such vectors (see for instance [11]).

Since we also want to deal with objects and operations of Euclidean geometry like circles and angular bisectors, we have to embed the usual Euclidean plane (equipped with a Euclidean metric) in $\mathbb{RP}^2$. A finite point $(x,y) \in \mathbb{R}^2$ will be represented by the point $(x,y,1)$ of $\mathbb{RP}^2$. With this standard embedding a line $ax+by+c=0$ of $\mathbb{R}^2$ is represented by (a,b,c), $\ell_\infty = (0,0,1)$ represents the line at infinity, and two lines $l_1 = (a_1,b_1,c_1)$ and $l_2 = (a_2,b_2,c_2)$ are orthogonal if $a_1b_2 - b_2a_2 = 0$. In order to simplify the notation later on we will also identify a finite point $(x,y,1)$ with a complex number $x+iy$. By this we identify the finite part of the projective plane with $\mathbb{C}$.

We restrict the use of angular bisectors to those lines that pass through the origin $(0,0)$ of $\mathbb{R}^2$. An angular bisector of two lines l_1 and l_2 through the origin is a line ℓ through the origin such that $\angle(l_1,\ell) = \angle(\ell,l_2)$. For a pair of lines there are two angular bisectors, which are orthogonal to each other. Restricting the use of angular bisectors to lines through the origin reduces the occurrence of non-admissible situations to a minimum. Formally, we will make use of the following primitive operations. For the sets P of points and L of lines, we define:

$$\begin{aligned}
\text{JOIN} &:= \{(p_1,p_2,l) \mid l \text{ is the line through } p_1 \text{ and } p_2 \text{ and } p_1 \neq p_2\} \subset (P\times P)\times L\\
\text{MEET} &:= \{\,(l_1,l_2,p)\ \mid p \text{ is the intersection of } l_1 \text{ and } l_2 \text{ and } l_1 \neq l_2\} \subset (L\times L)\times P\\
\text{BISECT} &:= \{\ (l_1,l_2,l)\ \mid l \text{ is an angular bisector of } l_1 \text{ and } l_2 \text{ and}\\
&\qquad\qquad\qquad\quad l_1,l_2,l \text{ pass through the origin}\} \qquad\qquad \subset (L\times L)\times L
\end{aligned}$$

Furthermore, we define the following four *constants* (i.e. primitives with input size zero):

$$P^{(a,b)} := \{(a,b,1)\} \subset P; \text{ for } a,b \in \{0,1\}.$$

These constants will be used to fix a coordinate system. For the generation of *free points* we define a special instruction that has no input elements and allows the output to be any point of P:

$$\text{FREE} := P.$$

We will deal with the following relational instruction set:

$$\text{JMB} := ((P,L),(\text{JOIN},\text{MEET},\text{BISECT},\text{FREE},P^{(0,0)},P^{(1,0)},P^{(0,1)},P^{(1,1)})).$$

Remark 2.3. Here are three comments on the choice of the primitive operations:

(i) The only cases where the two operations JOIN and MEET are not admissible is when the two input elements are identical. For all other cases they are well-behaved.

(ii) The only operation that introduces an ambiguity is BISECT. The primitives JOIN and MEET are "deterministic" in the sense that each admissible input has exactly one possible output.

(iii) The operation BISECT has been chosen for our investigations since it isolates the effect of generating an ambiguity. Unlike the *intersection circle with line* operation it has no open region of the input parameters where it is not admissible. Such effects (which we want to exclude here) would allow the possibility to construct some kind of "sidedness test", which are at the core of of complexity results for oriented matroids, polytopes, mechanical linkages or the warehouseman's problem. In Sec. 6 when we prove the PSPACE-hardness result we will make a very selected use of one such additional operation.

A construction sequence is formalized by the concept of a *geometric straight line program* (GSP).

Definition 2.4. A *geometric straight-line program* on a relational instruction set (O,Ω) is defined by a sequence of *statements* $\Gamma = (\Gamma_1,\dots,\Gamma_m)$. Each Γ_j has the form $\Gamma_j = (\omega, i_1,\dots,i_{\text{ar}(\omega)})$ where

(i) ω is an operation from the instruction set Ω,

(ii) the type of Γ_j is defined to be the type of ω,

(iii) for each $k \in \{1,\dots,\text{ar}(\omega)\}$ we have $i_k < j$,

(iv) for each $k \in \{1,\dots,\text{ar}(\omega)\}$ the type of Γ_{i_k} matches the type of the k-th input of ω.

After a suitable set of primitive operations is given it is straightforward to describe construction sequences by a GSP. Each statement $\Gamma_j = (\omega, i_1,\dots,i_{\text{ar}(\omega)})$ of a GSP describes the generation of a new element by means of a primitive operation ω whose input is given by the output of the statements $\Gamma_{i_1},\dots,\Gamma_{i_{\text{ar}(\omega)}}$. Item (iii) of the above definition ensures that only elements are used as input that have been already constructed. Item (iv) ensures a correct typing. The concept of a GSP emphasizes the constructive step-by-step nature, however it allows for a certain "non-determinism" during a construction, since it does not specify which output of an (ambiguous) operation to take. To make GSPs more readable we also use the " $\leftarrow$ " notation of Remark 2.2 to encode each statement. A statement $\Gamma_j = (\omega, i_1, \ldots, i_{\mathrm{ar}(\omega)})$ will then be written as $j \leftarrow \omega(i_1, \ldots, i_{\mathrm{ar}(\omega)})$. Furthermore, we allow to exchange the references $j, i_1, \ldots, i_{\mathrm{ar}(\omega)}$ by meaningful variable names.

We may consider a certain set of primitive operations as a kind of programming language. Each GSP is a certain program. In what follows we are mainly interested in the constructions/programs that can be described by the operations in JMB.

Example 2.5. *The following sequence of instructions is a simple GSP over the* JMB *instruction set. It takes two free points p and q, joins them to the origin o, and constructs the angular bisector of the two resulting lines.*

$$\begin{array}{rcl} p & \leftarrow & \textsc{free} \\ q & \leftarrow & \textsc{free} \\ o & \leftarrow & P^{(0,0)} \\ l_1 & \leftarrow & \textsc{join}(a,o) \\ l_2 & \leftarrow & \textsc{join}(b,o) \\ b & \leftarrow & \textsc{bisect}(l_1,l_2) \end{array}$$

We will still simplify the notions by assuming that points that do not occur explicitly on the left of any assignment are automatically initialized by a FREE operation. Furthermore, if the output of an operation is unique and used only once we allow that it is used directly (without intermediate variable) as an input of another operation. In particular this convention applies to the constants in JMB. With these conventions the above GSP can simply be written as

$$b \leftarrow \textsc{bisect}(\textsc{join}(p, P^{(0,0)}), \textsc{join}(q, P^{(0,0)})).$$

Closely related to the concept of a GSP $(\Gamma_1, \ldots, \Gamma_m)$ is the notion of an instance of the GSP. Roughly speaking an instance of a GSP is an assignment of a concrete object to each of the statements Γ_i such that all corresponding relations are satisfied.

Definition 2.6. An *instance* of a geometric straight-line program $(\Gamma_1, \ldots, \Gamma_m)$ is an assignment of objects $\tilde{X} = \tilde{X}_1, \ldots, \tilde{X}_m$ such that all primitives are *satisfied*, that is, for every statement $\Gamma_j = (\omega_j, i_1, \ldots, i_{\mathrm{ar}(\omega_j)})$ the relation $(\tilde{X}_{i_1}, \ldots, \tilde{X}_{i_{\mathrm{ar}(\omega_j)}}, \tilde{X}_j) \in \omega_j$ holds.

Example 2.7. *For the GSP given in Example 2.5. we have in particular the following instance (in homogeneous coordinates):*

$$\begin{array}{lll} \tilde{p} = (1,0,1) & \tilde{q} = (0,1,1) & \tilde{o} = (0,0,1) \\ \tilde{l}_1 = (1,0,0) & \tilde{l}_2 = (0,1,0) & \tilde{b} = (1,1,0) \end{array}$$

It is important that for the same choice of free elements there also exists another possible instance that exactly differs in the choice of the angular bisector:

$$\begin{array}{lll} \tilde{p} = (1,0,1) & \tilde{q} = (0,1,1) & \tilde{o} = (0,0,1) \\ \tilde{l}_1 = (1,0,0) & \tilde{l}_2 = (0,1,0) & \tilde{b} = (-1,1,0) \end{array}$$

Remark 2.8. By our definition of an instance we implicitly assume that for any specific instance the positions of the elements are *admissible* in the sense that each primitive operation can be executed.

Remark 2.9. A more formal treatment of RIS's and GSPs would include a careful separation of syntax and semantics of GSPs, a separation of references to objects and the objects themselves, and many other subtleties that are present whenever the aim is to formalize the concept of computing. However, we hope that the slightly informal treatment used in this article satisfies the needs of the reader as long as only complexity issues are concerned. A more elaborated treatment of GSPs can be found in [11].

2.2. **Continuity.** Along with the notion of GSPs and their instances comes a natural notion of continuity. For this we will split a specific GSP $P = (\Gamma_1, \dots, \Gamma_m)$ over the instruction set JMB into input variables and dependent variables. We consider each point in P that comes from a FREE operation as an *input* to P. W.l.o.g. we may assume that the definition of the input points are the first k statements P. Each of the operations JOIN,MEET, and BISECT has only a finite number of possible output values. This is the case since if we prescribe the positions of the input points all other objects of this instance are determined up to a finite number of possible binary choices. Each choice that has to be made comes from one application of a BISECT operation.

Now assume that $p_1, \dots, p_k$ are the input points of P. Furthermore, assume that we are given continuous functions

$$p_i(t)\colon [0,1] \to \mathbb{R}^3 \setminus \{(0,0,0)\}$$

for each $i \in \{1, \dots, k\}$. These functions describe a continuous movement of the input points (in homogeneous coordinates).

Definition 2.10. A *continuous evaluation* of the GSP P over the JMB instruction set under the movement $p_i(t)$ is an assignment of continuous functions

$$o_i(t)\colon [0,1] \to \mathbb{R}^3 \setminus \{(0,0,0)\}$$

for each $i \in \{k+1, \dots, m\}$ such that for all $t \in [0,1]$ the objects

$$(p_1(t), \dots, p_k(t), o_{k+1}(t), \dots, o_m(t))$$

form an (admissible) instance of P.

This concept formalizes the intuitive requirement that under a continuous movement of free elements the dependent elements should move continuously as well. For instance, if we have the simple GSP of Example 2.5 and move from one instance to another by changing the positions of the free elements a and b, a continuous evaluation makes sure that we do not jump spontaneously from one choice of the angular bisector to the other one.

Observe that the way we define continuity leaves room for the necessary indeterminism: Usually one would require that the output elements are given by continuous functions in the input, but here both the path of the input and the path of the output are given by continuous functions on the interval [0,1].

The following property of continuous evaluations is crucial:

Lemma 2.11. *If there exists a continuous evaluation of the GSP P over the* JMB *for a continuous movement $p_i(t)$ then it is unique.*

PROOF. We can prove this lemma by induction on the length of P. Assuming that the statement holds for all programs of length $m-1$ we prove that it also holds for programs of length m. Assume that for such a program P the functions $p_i(t)$ describe a continuous movement for which a continuous evaluation exists. If the last operation of P is one of the constant points then the statement holds trivially. If the last operation of P is one of the deterministic operations JOIN or MEET, then the statement holds by the continuity of these operations. If the last operation is BISECT then we can argue as follows: The two possible outputs of BISECT are two lines that are orthogonal to each other. If there was a way to continuously get from one branch to the other there must be a position in which these two lines coincide. This is impossible since the two angular bisectors are orthogonal. □

Remark 2.12. This Lemma shows the importance of non-admissible positions: At these *singularities* the different branches coincide, both angular bisectors degenerate to the zero vector. It is not possible to extend the projective setting by this additional line $(0,0,0)$ (and a corresponding point) without destroying the uniqueness of continuous evaluations, even though it is possible to extend the JMB instruction set to include them.

2.3. **Fundamental Problems in Dynamic Geometry.**

After formalizing the concept of GSPs and continuity we are finally in the position of formalizing the main questions of this work. The first problem formalizes the most fundamental operation of a Dynamic Geometry program: After you pick a free point of a construction and move it to another position, how did the rest of the construction change?

Definition 2.13. (Tracing problem): Let P be a GSP and let $p_i(t)$ describe a continuous movement for which a continuous evaluation $(p_1(t),\ldots,p_k(t),o_{k+1}(t),\ldots,o_m(t))$ exists. Furthermore, let $(p_1,\ldots,p_k,o_{k+1},\ldots,o_m)$ be an instance of P with free points $p_i = p_i(1)$ for all $i \in \{1,\ldots,k\}$. Decide whether $o_i = o_i(1)$ for all $i \in \{k+1,\ldots,m\}$.

The second problem asks for the mere existence of a path from one instance to another.

Definition 2.14. (Reachability problem): Let $P^0 = (p_1^0,\ldots,p_k^0,o_{k+1}^0,\ldots,o_m^0)$ and $P^1 = (p_1^1,\ldots,p_k^1,o_{k+1}^1,\ldots,o_m^1)$. Decide whether there exists a continuous evaluation that starts at P^0 and ends at P^1.

We will see that both problems turn out to be (at least) NP-hard. If we allow one single use of a sidedness test to constrain admissible regions the reachability problem even turns out to be PSPACE-hard.

3. Useful Gadgets

This section will describe small constructions that are helpful to compose the more complicated constructions that we need later.

3.1. **More Primitives.**

Since our set of primitive operations is very restricted we first show that other useful primitive operations can be easily composed from these primitives.

3.1.1. *The Line at Infinity.*

By a simple sequence of join and meet operations we can construct the line at infinity:

$$\begin{aligned}
a &\leftarrow \textsc{meet}(\textsc{join}(P^{(0,0)},P^{(0,1)}),\textsc{join}(P^{(1,0)},P^{(1,1)}))\\
b &\leftarrow \textsc{meet}(\textsc{join}(P^{(0,0)},P^{(1,0)}),\textsc{join}(P^{(0,1)},P^{(1,1)}))\\
\ell_\infty &\leftarrow \textsc{join}(a,b)
\end{aligned}$$

By construction, a and b are two distinct points on the line at infinity and hence ℓ_∞ is the line at infinity with homogeneous coordinates $(0,0,1)$.

3.1.2. *Parallel Lines.*

For a line l and a point p we can calculate the parallel to l through p by

$$\textsc{join}(\textsc{meet}(l,\ell_\infty),p).$$

If p lies on l this formula produces l itself. If $l = \ell_\infty$ this formula is not admissible. We will refer to this “macro” by $\textsc{parallel}(l,p)$.

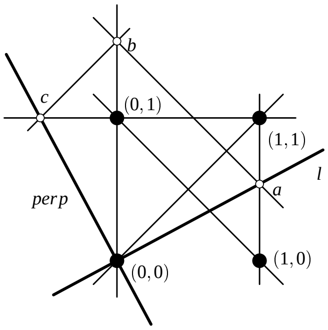


Fig. 3: Construction of a perpendicular.

3.1.3. *Perpendicular.* A bit less trivial is the construction of a perpendicular to l trough p. We can only do such a construction since the choice of our constant points provides us with a sample of two perpendicular lines. This right angle can then be transferred to another line. Since we already have a parallel operation w.l.o.g. we may assume that l passes through $P^{(0,0)}$ and that $p = P^{(0,0)}$. The construction is given in Fig. 3. We have

$$\begin{array}{rcl} a & \leftarrow & \mathrm{MEET}(\mathrm{JOIN}(P^{(1,0)},P^{(1,1)}),l), \\ b & \leftarrow & \mathrm{MEET}(\mathrm{JOIN}(P^{(0,0)},P^{(0,1)}),\mathrm{PARALLEL}(a,\mathrm{JOIN}(P^{(1,0)},P^{(0,1)}))), \\ c & \leftarrow & \mathrm{MEET}(\mathrm{JOIN}(P^{(0,1)},P^{(1,1)}),\mathrm{PARALLEL}(b,\mathrm{JOIN}(P^{(0,0)},P^{(1,1)}))), \\ perp & \leftarrow & \mathrm{JOIN}(c,P^{(0,0)}). \end{array}$$

This construction is admissible for all situations where l passes through $P^{(0,0)}$. We will refer to the general construction for perpendiculars by PERPENDICULAR(l,p).

3.2. **Arithmetics.** An essential part of our constructions will be the evaluation of certain polynomial expressions. For this we single out one particular line l on which we perform the evaluation. On this line we fix two points that play the roles of "0" and "1" and therefore fix an origin and a scale. To every point x on this line we can assign a unique value with respect to this scale. This value is given by the ratio $\frac{|0x|}{|01|}$ of oriented segment lengths. Sometimes we will abuse notation and use the name of the point as name for the value.

3.2.1. *Von Staudt Constructions.* The evaluation of arbitrary polynomials can be done if we are able to perform an elementary addition $z = x + y$ and an elementary multiplication $z = x \cdot y$. This can be done by the classical *von Staudt constructions*. They are shown in Fig. 4. In these pictures lines that seem to be parallel are *really* parallel. The desired arithmetic relations follow immediately from the similarities of the darkened triangles. Both constructions can be easily decomposed into a sequence of JOIN, MEET and PARALLEL

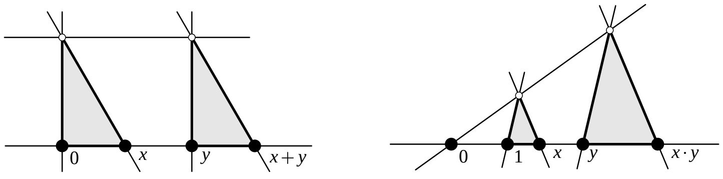


Fig. 4: Von Staudt constructions for addition and multiplication.

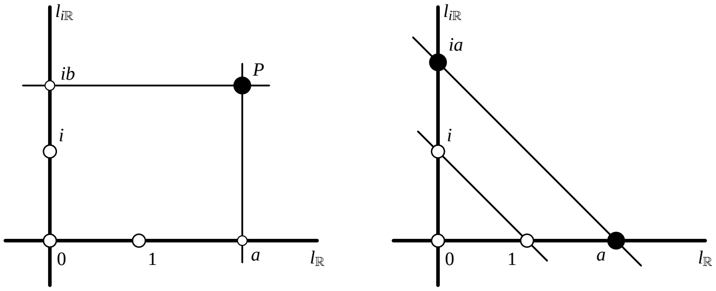


Fig. 5: Coordinate extraction for "complex" points.

operations, that start from the points 0, 1, x, y and one auxiliary point p not on l. In particular we get

$$\begin{aligned} x+y &\leftarrow \textsc{meet}(\textsc{join}(0,x),\\ &\qquad \textsc{parallel}(\textsc{join}(x,p),\\ &\qquad\qquad \textsc{meet}(\textsc{parallel}(\textsc{join}(0,p),y),\\ &\qquad\qquad\quad \textsc{parallel}(\textsc{join}(0,x),p)))),\\ x\cdot y &\leftarrow \textsc{meet}(\textsc{join}(0,x),\\ &\qquad \textsc{parallel}(\textsc{join}(x,p),\\ &\qquad\qquad \textsc{meet}(\textsc{parallel}(\textsc{join}(1,p),y),\textsc{join}(0,p)))). \end{aligned}$$

These construction sequences are chosen with care such that as long as the auxiliary point p is not on l the only non-admissible situations arise when in the addition both points x and y are at infinity or in the multiplication one of the points is at 0 and the other is at infinity.

3.2.2. *Complex Arithmetics.* As well as calculations over the real numbers we can also do calculations over *complex numbers*. For this we fix points "0", "1" and "i" in the plane, such that the lines $l_{\mathbb{R}} \leftarrow \textsc{join}(0,1)$ and $l_{i\mathbb{R}} \leftarrow \textsc{join}(0,i)$ are perpendicular and such that the distance from 0 to 1 is the same as the distance from 0 to i. For convenience we take $0 \leftarrow P^{(0,0)}$, $1 \leftarrow P^{(1,0)}$, $i \leftarrow P^{(0,1)}$. The lines $l_{\mathbb{R}}$ and $l_{i\mathbb{R}}$ play the roles of the real and imaginary axes of the complex plane. The points 0 and 1 define a scale on $l_{\mathbb{R}}$. The points 0 and i define a scale on $l_{i\mathbb{R}}$. For each point p in the plane we can (after orthogonal projection to these two axes) assign two coordinates, the real and the imaginary part of a complex number $a+ib$. If no confusion can arise we simply denote the points with homogeneous coordinates $(a,b,1)$ by the corresponding complex number $a+ib$. By a parallel projection along the direction of $\textsc{join}(1,i)$ we can transfer any number in $l_{i\mathbb{R}}$ to the corresponding number on $l_{\mathbb{R}}$, and vice versa. Let $z_1 = a_1 + ib_1$ and $z_2 = a_2 + ib_2$ be two complex numbers. With respect to our coordinate system we can model complex addition and complex multiplication of the points z_1 and z_2 by first transferring the real and imaginary parts to the line $l_{\mathbb{R}}$, then modeling the formulas

$$\begin{aligned} z_1 + z_2 &= (a_1+a_2) + i(b_1+b_2),\\ z_1 \cdot z_2 &= (a_1a_2 - b_1b_2) + i(a_1b_2 + b_1a_2), \end{aligned}$$

by a sequence of von Staudt constructions and finally construct a new point from the resulting real and imaginary part. The complex addition can be used for *addition of vectors* as well.

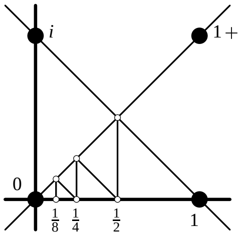


Fig. 6: Constructions for $\frac{1}{2^n}$.

Remark 3.1. One might think that using von Staudt constructions for vector addition is more than necessary. The simple construction sequence

$$z_1 + z_2 \leftarrow \text{MEET}(\text{PARALLEL}(0, z_1), \text{PARALLEL}(0, z_2))$$

seems to work as well. However, this construction has the disadvantage that it is non-admissible whenever 0, z_1 and z_2 are collinear. For the complexity issues that we consider later the actual length of these elementary operations is irrelevant as long as it is constant.

3.2.3. *Integer and Rational Points.* By being able to add and multiply via von Staudt constructions we are also able to construct points $a + ib$ for arbitrary integers a and b with respect to our coordinate system. We simply have to find a sequence of additions and multiplications that computes the numbers a and b starting from 0 and 1. In particular, using the binary representation any integer $n > 0$ can be constructed in $O(\log(n))$ construction steps.

It is also easy to construct numbers of the form $\frac{1}{2^n}$. The construction in Fig. 6 shows that this can be done in $O(n)$ steps.

3.3. **Points on Circles and Intervals.** In our relational instruction set JMB we do not have direct access to circles. However, by Thales' theorem we can freely generate points on circles that are given by two diameter points (see Fig. 7 left). Let a and b be the two endpoints of a diameter of the desired circle. We take a free point p and construct

$$q \leftarrow \text{MEET}(\text{JOIN}(a, p), \text{PERPENDICULAR}(\text{JOIN}(a, p), b)).$$

Using Thales' theorem it is immediate that the point p is on the circle with the segment $[ab]$ as diameter. We will abbreviate this construction by

$$q \leftarrow \text{ONCIRCLE}(a, b, p).$$

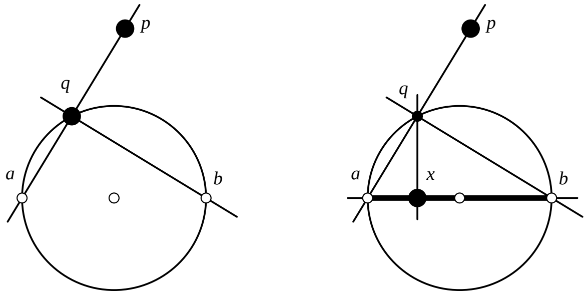


Fig. 7: Constructing points on circles and on segments.

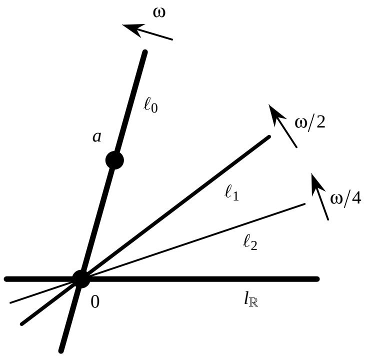


Fig. 8: Detection of a winding number.

If we furthermore project the resulting point orthogonally to the line a,b by

$$x \leftarrow \text{MEET}(\text{JOIN}(a,b),\text{PERPENDICULAR}(\text{JOIN}(a,b),\text{ONCIRCLE}(a,b,p))),$$

we get a point x that is constrained to lie in the closed segment from a to b (see Fig. 7 right). We abbreviate this by

$$x \leftarrow \text{ONINTERVAL}(a,b,p).$$

Only if p and a coincide these two operations are not admissible.

Remark 3.2. This construction has the side effect that while point p cycles once around point a, the derived point q makes *two* full cycles on the circle.

Remark 3.3. Although by this construction we can generate a point freely on the boundary of a circle, this circle is not available for further constructions like intersecting it with a line.

3.4. **Detecting a Winding Number.** So far the constructions used in our gadgets did not contain any BISECT operations and therefore no non-determinism occurred. The basic functionality for which we will use BISECT operations is the generation of *monodromy effects*: "One can start with an instance A of a GSP and continuously make a round-trip with the free elements and end up in a different instance B." The smallest device for which such an effect occurs is given by the following GSP:

$$\ell_0 \leftarrow \text{JOIN}(a,0)$$
$$\ell_1 \leftarrow \text{BISECT}(l_{\mathbb{R}},\ell_0)$$
$$\ell_2 \leftarrow \text{BISECT}(l_{\mathbb{R}},\ell_1)$$

It takes a free point a, joins it with the origin and constructs an angular "quad-sector" of this line and $l_{\mathbb{R}}$. Assume that a is in a certain position ($\neq 0$) and from there makes a round-trip with continuous speed around the origin. While ℓ_0 moves with an angular velocity ω the line ℓ_1 moves with angular velocity $\omega/2$ and the line ℓ_2 moves with angular velocity $\omega/4$. Thus after a has performed a full cycle around the origin the line ℓ_2 has made a quarter turn. If a moves along an arbitrary path (that avoids point 0) and returns to its original position then the resulting situation reflects the parity of the winding number of a around the origin. If ℓ_2 returned to its original position, the winding number was even, if ℓ_2 moved to the orthogonal of its original position the winding number was odd.

We can furthermore iterate this construction by adding more statements of the form

$$\ell_i \leftarrow \text{BISECT}(l_{\mathbb{R}},\ell_{i-1})$$

for $i \in \{3,\dots,k\}$. The resulting line ℓ_k moves with angular velocity $\omega/2^k$. By this construction we can determine the winding number of a round-trip of a modulo the exponential number 2^{k-1}. If the line ℓ_k made a total turn of $i\cdot\pi/2^{k-1}$ the winding number w satisfies

$$w \equiv i \mod 2^{k-1}.$$

The situation for the first two iterations is shown in Fig. 8.

4. Reachability Problems

This chapter is dedicated to our first theorem. We will prove:

Theorem 4.1. *The following decision problem is NP-hard: Given a GSP $\mathcal{P}$ over the* JMB *instruction set that uses at most three* BISECT *operations. Furthermore, given two instances A and B of $\mathcal{P}$. Decide whether there is an admissible real path from A to B.*

We will prove this theorem by giving a reduction from the well known 3-SAT decision problem.

4.1. From 3-SAT to Algebra.

The following problem is one of the standard NP-complete decision problems [2].

Decision Problem 4.2 (3-SAT). *Let $B = (b_1,\dots,b_n)$ be boolean variables, and let the literals over B be $\widetilde{B} = (b_1,\dots,b_n,\neg b_1,\dots,\neg b_n)$. Furthermore, let $C_1,\dots,C_k$ be clauses formed by disjunction of three literals from $\widetilde{B}$. Decide whether there is a truth assignment for B that satisfies all clauses $C_1,\dots,C_k$ simultaneously.*

W.l.o.g. we may assume that each variable occurs at most once in each clause. We first give a (polynomial time) procedure that transfers each instance of 3-SAT into a corresponding problem concerning the roots of a multivariate polynomial. Let $b_1,\dots,b_n$ be the boolean variables and let $C_1,\dots,C_k$ be the clauses of a given 3-SAT S. To each b_i we assign a formal variable x_i. For a literal $l_i \in \{b_i,\neg b_i\}$ we set

$$f(x_i) := \begin{cases} x_i & \text{if } l_i = b_i, \\ 1-x_i & \text{if } l_i = \neg b_i. \end{cases}$$

Assume that for each $j = 1,\dots,k$ the clause C_j is of the form $l_r^j \vee l_s^j \vee l_t^j$ where the literal l_i^j is either b_i or $\neg b_i$. We set

$$F_j := f(l_r^j)\cdot f(l_s^j)\cdot f(l_t^j).$$

Finally we set

$$F_S = \sum_{j=1}^{k} F_j.$$

By this translation for instance the 3-SAT formula $(b_1 \vee \neg b_3 \vee b_5) \wedge (\neg b_2 \vee b_4 \vee \neg b_5)$ is translated to $(x_1\cdot(1-x_3)\cdot x_5) + ((1-x_2)\cdot x_4\cdot(1-x_5))$. The satisfying truth assignments for S and the roots of F_S in $[0,1]^n$ are related by the following lemma (here $[0,1]$ denotes the closed interval between 0 and 1).

Lemma 4.3. *S has a satisfying truth assignment if and only if there are $(x_1,\dots,x_n) \in [0,1]^n$ with $F_S(x_1,\dots,x_n) = 0$.*

Proof. If S has a satisfying truth assignment $(b_1,\dots,b_n) \in \{\text{True},\text{False}\}^n$ we set

$$x_i := \begin{cases} 0 & \text{if } b_i = \text{True}, \\ 1 & \text{if } b_i = \text{False}. \end{cases}$$

Since every clause contains at least one true literal we the get that all $f_1,\dots,f_k$ are zero. This yields that F_S is zero as well. Conversely, assume that there are values $(x_1,\dots,x_n) \in$

$[0,1]^n$ such that $F_S(x_1,\dots,x_n) = 0$. If the x_i are chosen in the interval $[0,1]$ all f_j are non-negative. Thus $\sum_{j=1}^k f_j = 0$ implies that all f_j are zero. However, each f_i can only be zero if at least one of its factors is zero. By setting

$$b_i := \begin{cases} \text{TRUE} & \text{if } x_i = 0, \\ \text{FALSE} & \text{if } x_i \neq 0, \end{cases}$$

we get a satisfying truth assignment for S. □

Using the structure of the polynomial $F_S(x_1,\dots,x_n)$ we can derive a simple gap theorem in the case that S is not satisfiable.

Lemma 4.4. *If S is not satisfiable then $F_S(x_1,\dots,x_n) \geq 1$ for all $(x_1,\dots,x_n) \in [0,1]^n$.*

PROOF. This is true since F_S is a multilinear form and $F_S(x_1,\dots,x_n)$ is an integer that is greater or equal to zero for all vertices $(x_1,\dots,x_n) \in \{0,1\}^n$ of the unit cube. □

4.2. **From Algebra to Geometry.** Our next step is to transfer the algebraic situation in F_S to a geometric construction using exclusively JOIN and MEET operations and the constant points 0, 1, i, and $1+i$. This construction has the following properties: It contains freely movable points $p_1,\dots,p_n$ (one for each boolean variable in S), and a dependent point q that is constrained to lie on $l_{\mathbb{R}}$. There will be admissible positions for $p_1,\dots,p_n$ such that 0 and q coincide if and only if S is satisfiable.

Using the gadgets from Sec. 3 the construction is straightforward. We construct n points $x_1,\dots,x_n$ according to

$$x_i \leftarrow \text{ONINTERVAL}(0,1,p_i).$$

The construction constrains each of the points x_i to the segment $[0,1]$ (see Sec. 3.3). Except for this there is no restriction to the positions of the points $x_1,\dots,x_n$. These points model the input variables $x_1,\dots,x_n$ of the equation F_S whose values should be chosen in the interval $[0,1]$.

Using von Staudt constructions we now encode the polynomial $F_S(x_1,\dots,x_n)$ geometrically. All calculations are carried out on the line $l_{\mathbb{R}}$. The point that finally represents the result of the calculation is called q. This point q lies on $l_{\mathbb{R}}$ by its construction and Lemma 4.3. It can coincide with 0 if and only if S was satisfiable. We call the whole construction C_S. Altogether we obtain:

Lemma 4.5. (i) *In C_S the point q lies on $l_{\mathbb{R}}$.*
(ii) *There is an admissible position for $p_1,\dots,p_n$ in C_S such that q and 0 coincide if and only if S has a satisfying truth assignment.*
(iii) *If S is not satisfiable then $q \geq 1$ for all admissible positions of $p_1,\dots,p_n$.*

PROOF. (i) The point q lies on $l_{\mathbb{R}}$ by construction. (ii) is a consequence of the construction and Lemma 4.3. (iii) is a consequence of the construction and Lemma 4.4. □

4.3. **A Geometric Combination Lock.** Our final task for proving Thm. 4.1 is to transfer the construction C_S into a construction that can be used for proving NP-hardness of a reachability problem. The construction of C_S so far included only JOIN and MEET operations without non-determinism. The idea now is to conclude the construction by linking it to the "winding number gadget" presented in Sec. 3.4. We will do this in such a way such that a certain angular bisector can be rotated by $\pi/2$ if and only if the original 3-SAT problem was satisfiable.

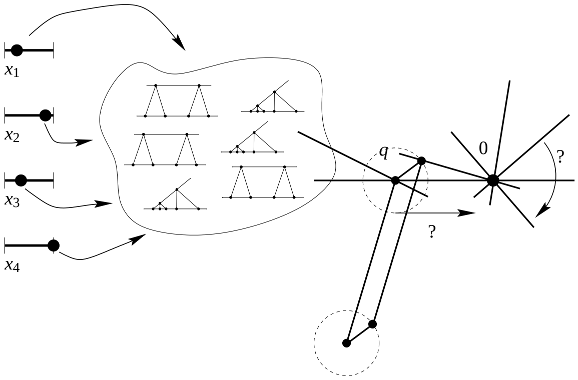


Fig. 9: Schematic view of the construction of a "geometric combination lock".

For this we add a new free point p from which we construct a derived point v on the circle with diameter $[-\frac{1}{2}+0i, \frac{1}{2}+0i]$. The construction is standard using the point on circle gadget from Sec. 3.3:

$$v \leftarrow \text{ONCIRCLE}(-\frac{1}{2}, \frac{1}{2}, p).$$

Then we take the construction for C_S and use our gadget for complex addition to construct $w = q + v$ and the line $\ell_0 \leftarrow \text{JOIN}(0, w)$. Finally, we add three non-deterministic statements

$$\ell_1 \leftarrow \text{BISECT}(l_{\mathbb{R}}, \ell_0), \quad \ell_2 \leftarrow \text{BISECT}(l_{\mathbb{R}}, \ell_1), \quad \ell_3 \leftarrow \text{BISECT}(l_{\mathbb{R}}, \ell_2).$$

The line ℓ_3 is a three times iterated angular bisector of $l_{\mathbb{R}}$ and ℓ_0. By construction the lines ℓ_0, ℓ_1, and ℓ_2 pass through the origin. Hence the bisector operations are admissible. If the positions of $p, p_1, \ldots, p_n$ are fixed then the construction is completely determined up to the actual position of ℓ_1, ℓ_2, and ℓ_3. The final construction is called R_S.

For the positions $\widehat{p} = \widehat{p}_1 = \ldots = \widehat{p}_n = 1$ the line ℓ_0 coincides with $l_{\mathbb{R}}$ and the choice

$$\ell_1 = \ell_2 = \ell_3 = l_{\mathbb{R}}$$

is a proper instance A of R_S. For these positions of $p, p_1, \ldots, p_n$ also the choice

$$\ell_1 = \ell_2 = l_{\mathbb{R}}, \quad \ell_3 = l_{i\mathbb{R}}$$

is a proper instance B. The only distinction between the instances A and B is the position of ℓ_3.

Lemma 4.6. *There is a continuous admissible path from A to B (induced by a movement of the free points $p, p_1, \ldots, p_n$) if and only if S is satisfiable.*

PROOF. The only way to get from A to B is that the point $w = q + v$ turns an odd number of cycles around the origin. Assume for a moment that $p_1, \ldots, p_n$ are fixed. Then the point q has a certain position on the line $l_{\mathbb{R}}$. The point w is then constrained to lie on a circle of radius $\frac{1}{2}$ around q. By moving p we can freely influence the position of w on this circle. The only way to let w cycle around the origin is to move p to a position that has less than distance $\frac{1}{2}$ to the origin, and then move p to achieve a full cycle of w around the origin. However, Lemma 4.5 shows that q can only come so close to the origin if and only if S was satisfiable. This proves the claim. □

We may think of the whole construction as a "geometric combination lock:" The points $p_1, \ldots, p_n$ play the role of the code dials. The point p plays the role of an opening wheel.

The angular bisector is the bolt of the combination lock. The reachability problem translates to the question whether one can open the lock. Initially the dials and the wheel are in some position. If we want to open the combination lock we first have to move the dials into the correct position (this can only be done if we know the solution to the 3-SAT problem S). If the dials are in the correct position we can turn the opening wheel and open the lock. After opening the lock we move all dials and the opening wheel again to the initial position. Nothing has changed except for the fact that the lock is now open.

A schematic picture of the whole situation is shown in Fig. 9. Points on an interval are used for von Staudt constructions. The result of this computation is used for the opening wheel.

Finally, observing that the whole translation from the original 3-SAT to the construction $\mathcal{R}_S$ can be carried out in polynomial (even linear) time in the length of the 3-SAT problem proves Thm. 4.1.

5. Computing a Specific Trace

The goal of this section is to prove our next main theorem. It describes the complexity of the basic situation in a Dynamic Geometry system: You pick a point and move it from one position to another. It will turn out to be NP-hard to decide whether a continuous evaluation of the situation ends up in a specific situation.

Theorem 5.1. *Given a GSP $\mathcal{P}$ over the* JMB *instruction set that contains exactly one free point p. Furthermore, given two instances A and B such that p is at position a in A and p is at position b in B. Let $p(t) : [0,1] \to [a,b]$ be a (straight) movement of p with $p(0) = a$ and $p(1) = b$. It is NP-hard to decide whether a continuous evaluation of $\mathcal{P}$ under this movement that starts at instance A ends up at the instance B.*

Here is an overview over the ingredients of our proof: First, we map the moving point p to the unit circle. Then we construct a set of polynomials $B_j(z)$ that correspond to the variables of a given 3-SAT problem in a way that all possible 0-1 combinations are represented by the values of the B_j on the unit circle. Finally, another polynomial $F_s(z)$ encodes the boolean formula of the 3-SAT problem and controls a point q, that will cycle around the origin. The winding number of this point can be used to read off the satisfiability of the 3-SAT.

A similar polynomial construction has been used by Plaisted in [17, 18, 19]. He used it to prove that it is NP-hard to decide for a sparse univariate polynomial whether it has a complex root of modulus 1. Our constructions differ from Plaisted's work by being more focused on evaluations of polynomials over the real numbers. One of the direct consequences of our construction is that it is NP-hard to decide whether a real polynomial encoded by a straight line program has a root over the real numbers (see Sec. 5.7). This fact can also be derived as a consequence of Plaisted's Theorem by a Moebius transformation argument. The alert reader will find out that we could have used the binary counter construction of Sec. 6 to prove Thm. 5.1, but the additional results for real polynomial roots (and some additional insight) would not have been possible then. We are convinced that the additional effort pays off very well.

5.1. A Point on the Unit Circle.

We will start our construction with a little gadget that maps a certain line segment to a point on the unit-circle. For this we first use the point-on-circle gadget of Sec. 3.3. and set

$$w \leftarrow \textsc{oncircle}(-1, 1, p).$$

If p is located at 2 the point w is located at 1. While p moves on a straight vertical path to the point $2+3i$ point w makes a quarter turn on the unit circle. We set

$$z \leftarrow w^4.$$

This point is constructible by the gadgets for complex arithmetics from Sec. 3.2. While p moves along the segment from 2 to $2+3i$ the point $z =: z(p)$ makes exactly one full cycle on the unit circle.

5.2. **Complex Polynomials for Variables.** From now on we fix a specific instance S of a 3-SAT problem with variables $b_1,\dots,b_n$ and clauses $C_1,\dots,C_k$. We will encode S into an instance of the decision problem of Thm. 5.1.

Let P_j be the j-th prime number, and let $M = \prod_{j=1}^{n} P_j$ be the product of the first n primes. The size of the j-th prime is less than $j\log(j)$. Hence the size of M is less than $n^{n\log n}$. The polynomial $z^M - 1$ has altogether M single roots, the M-th roots of unity, equally spaced on the unit circle at $z = e^{2i\pi(r/M)}$, for $r \in \{1,2,\dots,M\}$. We abbreviate $\varepsilon_M(r) = e^{2i\pi(r/M)}$. We consider two classes of polynomials for which the sets of roots are subsets of the roots of $z^M - 1$:

$$\begin{array}{rcl} B_j(z) & = & 1 - z^{M/P_j}, \\ \overline{B}_j(z) & = & 1 + z^{M/P_j} + z^{2M/P_j} + z^{3M/P_j} + \cdots + z^{(P_j-1)M/P_j}. \end{array}$$

For each $j \in \{1,\dots,n\}$ we set $A_j = \{P_j, 2P_j, 3P_j, \dots, M\}$ and $\overline{A}_j = \{1,2,\dots,M\} - A_j$. The following relations are immediate.

Lemma 5.2. *With the notation as set above we have:*

(i) *For each $j \in \{1,\dots,n\}$ we have $B_j(z)\cdot\overline{B}_j(z) = z^M - 1$.*
(ii) *The roots of $B_j(z)$ are at $z = \varepsilon_M(r)$, for $r \in A_j$.*
(iii) *The roots of $\overline{B}_j(z)$ are at $z = \varepsilon_M(r)$, for $r \in \overline{A}_j$.*

PROOF. Claim (i) can be directly proved by expansion of the product. Claim (ii) is trivial. Claim (iii) is a consequence of (i) and (ii) and the fact that there are no multiple roots in $z^M - 1$. □

Later on we will associate to each number $r \in \{1,2,\dots,M\}$ a certain evaluation of the boolean variables $b_1,\dots,b_n$. For a number r the boolean variable b_j will be considered TRUE if $\varepsilon_M(r)$ is a root of B_j and FALSE otherwise. The above lemma proves that under this correspondence b_j is FALSE (at r) if and only if $\varepsilon_M(r)$ is a root of $\overline{B}_j$. We set

$$b_j(r) := \begin{cases} \text{TRUE} & \text{if } B_j(\varepsilon_M(r)) = 0, \\ \text{FALSE} & \text{if } B_j(\varepsilon_M(r)) \neq 0. \end{cases}$$

Lemma 5.3. *For each truth assignment $(b_1,\dots,b_n) \in \{\text{TRUE},\text{FALSE}\}^n$ there is at least one number $r \in \{1,\dots,M\}$ such that $b_j(r) = b_j$ for all $j \in \{1,\dots,n\}$.*

PROOF. For a given assignment $(b_1,\dots,b_n) \in \{\text{TRUE},\text{FALSE}\}^n$ we are looking for an integer $r \leq M$ that has P_j as a prime factor if and only if $b_j(r) = \text{TRUE}$. For this we can simply take the number

$$\prod_{\{j \mid b_j(r)=\text{TRUE}\}} P_j$$

which has this property. □

We will explicitly calculate the above polynomials $B_j(z)$, $\overline{B}_j(z)$ and $z^M - 1$ by suitable geometric constructions. We will have to take care that the effort of doing this is no more than polynomial in the coding length of the original 3-SAT S. For this we need:

Lemma 5.4. *For each n the polynomial z^n can be evaluated by a straight line program using at most $O(\log_2(n))$ multiplications. Similarly for each n and each divisor m of n the polynomial $1 + z^m + z^{2m} + \cdots + z^n$ can be evaluated by a straight line program using at most $O(\log(n)^2)$ additions or multiplications.*

PROOF. Let $n = \sigma_0 2^0 + \sigma_1 2^1 + \sigma_2 2^2 + \cdots + \sigma_k 2^k$ with $k \leq \log_2(n)$ and $\sigma_i \in \{0,1\}$ be the binary expansion of n. We can write $z^n = \prod_{\{i \mid \sigma_j = 1\}} z^{2^j}$. This product has at most $\log_2(n)$ terms. The polynomial $z^{2^j} = z^{2^{j-1}} \cdot z^{2^{j-1}}$ uses only one additional multiplication if we already have $z^{2^{j-1}}$. This proves the first claim.

For the second claim we first set $f_k(z) = 1 + z^1 + z^2 + \cdots + z^k$. We have $1 + z^m + z^{2m} + \cdots + z^n = f_{n/m}(z^m)$. Thus after having used $2\log_2(m)$ multiplications for computing z^m we just have to care about $f_k(z)$ for $k = n/m$. If k is even we get $f_k(z) = f_{k/2}(z)(1 + z^{k/2})$, if k is odd we get $f_k(z) = f_{k/2-1}(z)(1 + z^{k/2}) + z^{k/2}$, and a simple recursion on k proves the claim. □

The last lemma together with the observation that M is less than $n^{n \log n}$ shows that all polynomials $B_j(z)$, $\overline{B}_j(z)$ and $z^M - 1$ can be encoded by straight line programs whose length is polynomial in n.

5.3. **Evaluating a 3-SAT.** We now proceed by encoding the original 3-SAT instance S into our construction. For a complex number $z = a + ib$ we set $||z|| = a^2 + b^2$. For all $j \in \{1, \ldots, n\}$ we consider $L_j(z) = ||B_j(z)||$ and $\overline{L}_j(z) = ||\overline{B}_j(z)||$. These two functions are *real-valued* and even non-negative. The only way for these functions to be zero is that the corresponding functions $B_j(z)$ and $\overline{B}_j(z)$ become zero. For a literal $l_j \in \{b_j, \neg b_j\}$ we set

$$f^{l_j}(z) := \begin{cases} L_j(z) & \text{if } l_j = b_j, \\ \overline{L}_j(z) & \text{if } l_i = \neg b_i. \end{cases}$$

Assume that for each $j = 1, \ldots, k$ the clause C_j is of the form $l_r^j \vee l_s^j \vee l_t^j$ where the literal l_k^j is either b_k or $\neg b_k$. We set

$$F_j(z) := f^{l_r^j}(z) \cdot f^{l_s^j}(z) \cdot f^{l_t^j}(z).$$

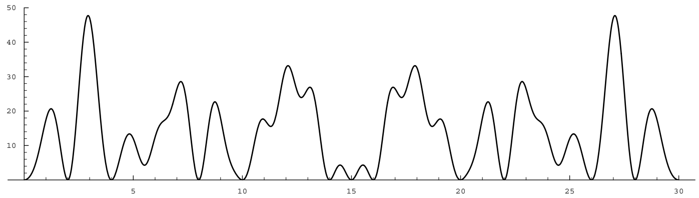
Finally we set

$$F_S(z) = \sum_{j=1}^{k} F_j(z).$$

If $z = a + ib$ the function $F_S(z)$ is a real polynomial in a and b that can be realized by a straight line program with length polynomial in the size of the 3-SAT instance S. It is important that $F_S(z)$ is not an element of the polynomial ring $\mathbb{C}[Z]$, because otherwise the following lemma could not be true.

Lemma 5.5. *It is NP-hard to decide whether there is a $z \in \mathbb{C}$ with $F_S(z) = 0$.*

PROOF. The only way for $F_S(z)$ to become zero is that all its summands are zero. This however can only be the case if z is of the form $\varepsilon_M(r)$ for an r that corresponds to a satisfying truth assignment of S. □



1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1

1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1

Fig. 10: The graph of the function $F_S(\cos(2\pi t/30) + i\sin(2\pi t/30))$.

Example 5.6. Let us consider a specific satisfiability problem S and the associated function F_S. In order for the example to have a reasonable size we consider a 2-SAT instead of an actual 3-SAT instance:

$$S = (b_1 \vee b_3) \wedge (b_2 \vee b_1) \wedge (\neg b_2 \vee b_3).$$

The satisfying truth assignments are $(1,0,0)$, $(1,0,1)$ and $(1,1,1)$. We associate b_1 with the prime number 2, b_2 with 3, and b_3 with 5. The resulting graph of $F_S(\cos(2\pi t/30) + i\sin(2\pi t/30))$ together with the corresponding bit patterns is shown in Fig. 10. The ticks mark the corresponding 30^{th} roots of unity. Whenever we have a bit pattern corresponding to a satisfying assignment the function is zero, else it is greater than zero.

5.4. **Another Gap-Theorem.** Again we need an estimate how small the function $F_S(z)$ can become if S is not satisfiable. In fact we will need this minimum only for points $\varepsilon_M(r)$ with $r \in \{1,2,\ldots,M\}$. We can get a lower bound for this value by calculating the smallest possible non-zero summand of $F_S(\varepsilon_M(r))$. This value in turn can be bounded by the cube of the smallest non-zero value α that one of the functions $L_j(\varepsilon_M(r))$, $\overline{L}_j(\varepsilon_M(r))$ can take for $r \in \{1,2,\ldots,M\}$. For this it is useful to observe that $L_j(\varepsilon_M(r)) = 2 - 2\cos(2\pi \cdot r/P_j)$. This desired value α is taken at $L_n(\varepsilon_M(1))$. Thus we have $\alpha = 2 - 2\cos(2\pi/P_n)$. We obtain the following lemma.

Lemma 5.7. *If $F_S(\varepsilon_M(r))$ is non-zero for some $r \in \{1,\ldots,M\}$, then we have*

$$F_S(\varepsilon_M(r)) \geq (2 - 2\cos(2\pi/P_n))^3 > (2 - 2\cos(2\pi 2^{-\log(n)^2})^3.$$

PROOF. The first inequality follows from our considerations above. The second inequality is a very rough estimate following from the monotonicity of $\cos(t)$ in $[0,\pi/2]$ and the fact that $P_n < 2^{\log(n)^2}$. □

We set $\beta_S = (2 - 2\cos(2\pi \cdot 2^{-\log(n)^2}))^3$. This number can be constructed geometrically using an iterative sequence of $\log(n)^2$ BISECT operations starting with the right angle followed by a constant number of JOIN and MEET operations.

5.5. **Tracing the Flight of a Bumble Bee.** Now we are done with the algebraic part of our construction. We come back to the geometric part that started with the construction of a point z that moves once around the unit circle while the free point p moves from a to b (Sec. 5.1). We take z as the input of $F_S(z)$ and model the evaluation of $F_S(z)$ by von Staudt constructions as described in Sec. 3.2. The result is a non-negative point q on the real axis $l_{\mathbb{R}}$. This point can coincide with the origin whenever z corresponds to a satisfying truth

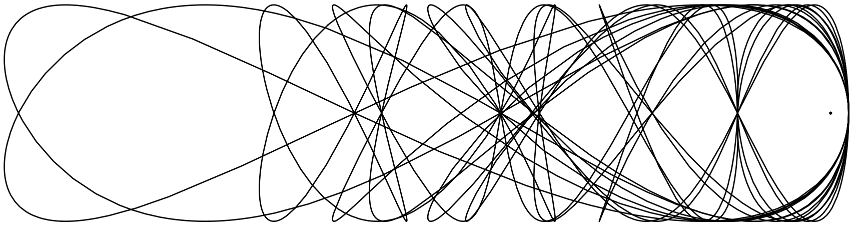


Fig. 11: The path of the dependent point $g_S(z)$.

assignment of S. In particular, this can only happen if $z = \varepsilon_M(r)$ for a suitable r. We now consider the function

$$g_S(z) := z^M - 1 - F_S(z) + \beta_S.$$

Lemma 5.8. *Let G be the path of $g_S(z)$ while z makes one full cycle around the unit circle. The winding number of G with respect to the origin is zero if and only if S has no satisfying truth assignment.*

PROOF. Let $z = e^{2i\pi t}$ for $t \in [0,1]$ move with constant speed on the unit circle. We can get the corresponding winding number around the origin in the following way: We take the ray of all positive real numbers. We calculate two numbers: w^+ counts how often the point $g_S(z)$ crosses this ray moving from the lower to the upper half plane, and w^- counts how often the point $g_S(z)$ crosses this ray moving from the upper to the lower half plane. The number $w^+ - w^-$ gives the winding number.

The real part of $g_S(z)$ is by construction and by Lemma 5.7 at most β_S. For $g_S(z)$ being real and positive the numbers $z^M - 1$ and $F_S(z)$ must both vanish. Thus we obtain

$$w^+ = |\{r \in \{1, \ldots, M\} \mid F_S(\varepsilon_M(r)) = 0\}| \quad \textit{and} \quad w^- = 0.$$

The winding number counts exactly the number of possible $r \in \{0, \ldots, M\}$ for which $b_j(r)$ is a satisfying truth assignment of S. □

Fig. 11 shows the trace of $g_S(z)$ for the 2-SAT formula of Example 5.6. The origin lies on the symmetry axis of the figure very close to the right boundary. The winding number will be exactly 12 corresponding to the 12 zeros of $F_S(z)$ shown in Fig. 10.

5.6. **NP-hardness of Tracing.** Now we are in principle done. We do a geometric construction that calculates $g_S(z(p))$ using the free point p as input parameter. This can be done by a number of JOIN, MEET, and BISECT operations that is polynomial in the parameter n and k of S (by Lemma 5.4 and the considerations in Sec. 5.4). We construct the line $\ell_0 = \text{JOIN}(0, g_S(z(p)))$. Finally, we add n^3 non-deterministic statements:

$$\begin{array}{rcl} \ell_1 & \leftarrow & \text{BISECT}(l_{\mathbb{R}}, \ell_0) \\ \ell_2 & \leftarrow & \text{BISECT}(l_{\mathbb{R}}, \ell_1) \\ & \vdots & \\ \ell_{n^3} & \leftarrow & \text{BISECT}(l_{\mathbb{R}}, \ell_{n^3-1}) \end{array}$$

The winding number w of Lemma 5.8 satisfies

$$w < M < n^{n \log(n)} < n^{n^2} < (2^n)^{n^2} = 2^{n^3}.$$

Thus we obtain the following lemma.

Lemma 5.9. *Let A be an initial position of our entire construction for $p = 2$ and let B be the corresponding position with identical choice of the angular bisectors for $p = 2+3i$. B is the result of a continuous evaluation under the straight movement of p if and only if S was not satisfiable.*

PROOF. If S is not satisfiable, then the winding number of $g_S(z(p))$ around the origin is zero and the position of the angular bisectors remains unchanged. If S is satisfiable, then the winding number lies between 1 and 2^{n^3}. Thus a movement of p causes a change of the positions of the angular bisectors. □

This concludes our proof of Thm. 5.1, which is a direct consequence of the above Lemma and the fact that the construction was polynomial in the size of n and k.

5.7. **Roots of Univariate Polynomials.** We will close this chapter with a little side remark. Assume that the free point p of our construction is parameterized by $(2,x,1)$ (in homogeneous coordinates with $x \in \mathbb{R}$). The construction of the function $F_S(z(p))$ could be exclusively done with JOIN and MEET operations. The coordinates of the resulting point $F_S(z(p)) = (\alpha(x),\beta(x),\gamma(x))$ are then polynomials in the single variable x. These polynomials can be calculated by straight line programs whose length is polynomial in the size of S, by translating the GSP into an equivalent SLP (see [11]). The 3-SAT S is satisfiable if and only if there is an x with $\alpha(x) = 0$. Thus we obtain

Theorem 5.10. *It is NP-hard to decide whether a univariate polynomial encoded by a straight line program has a real root.*

6. PSPACE-HARD PROBLEMS

This section focuses on proving that certain reachability problems are PSPACE-hard do decide. Compared to the previous sections there is one important difference. Every construction done so far only needed *constant points*, *meet*, *join*, and *bisect* operations. For the proof of the following PSPACE-hardness results for real reachability we need one additional ingredient, a semialgebraic constraint on the configuration space of the geometric configuration. We will demonstrate several variants of the result with different such constraints:

- the condition that a certain point is always on the left of a certain line,
- the condition that a certain point is always inside a certain circle,
- the condition that the intersection of a line and a circle is always real,
- the condition that the total length of the path of a freely movable point stays below a certain threshold.

Note that the first three variants can be transformed into each other. In fact there are many other variants of the result since the necessary restrictions that come from the additional inequality can be formalized in a very weak way. Nevertheless we have not been able to derive a comparable result without the additional condition. Our proof will be entirely constructive and self contained. It just relies on the well known PSPACE hardness of *quantified boolean formulas*. Moreover we will be very restrictive in the use of free points: the final construction has only *one* free point.

Let us first formulate one of the natural version of the main result of this section which is very similar to Thm. 4.1, except for the additional inequality constraint (for which we choose an *incircle test* here).

Theorem 6.1. *The following decision problem is PSPACE-hard: Given a GSP $\mathcal{P}$ over the* JMB *instruction set that has exactly one free point and a certain dependent point d. Furthermore, given two instances A and B of $\mathcal{P}$. Decide whether there is an admissible real path from A to B, such that along the path we always have $|d| < 2$.*

The proof of this result will be done by a reduction to the PSPACE-hardness of *Quantified Boolean Formulas* (QBF). Formally the PSPACE-hardness of QBF can be stated as "it is PSPACE-hard to decide whether the formula

$$\forall x_1 \exists y_1 \forall x_2 \exists y_2 \dots \forall x_n \exists y_n f(x_1, y_1, x_2, y_2, \dots, x_n, y_n)$$

is true, where f is a boolean expression." In a sense this formula resembles a two player game with players X and Y. It asks for a winning strategy for player Y. The formula f encodes the winning positions of Y: "For each move x_1 of X there is a move y_1 for Y such that for each move x_2 of X there is a move y_2 for Y such that … such that there is a final move y_n for Y such that Y wins the game.

For the proof we will geometrically construct a binary counter that counts through all possibilities for the $x_1, \dots, x_n$. The entire construction is such that in order to get from position A to B in the reachability problem one has (at certain positions) to *set* the values of the $y_1, \dots, y_n$ properly which can only be done if one knows the complete strategy for player Y.

Remark 6.2. Before we start with the proof let us contemplate for a moment the value of the following constructions. It is a remarkable fact that a similar result can be obtained even without using the BISECT operation at all – however for the price of an unbounded number of free points. The idea for this is to use one of the well known PSPACE-hard semialgebraic reachability problems (like the warehouseman's problem [7, 24]) as the starting point of the reduction. All involved equations and inequalities can be condensed into one big inequality (this new inequality describes an ε-approximation of the original problem). This translation can only be done with the help of additional slack-variables (one variable per original inequality). The information of a certain state of the construction is "stored" in the actual values of the slack variables. Particular technical difficulties arise from the right choice of the involved ε-sizes.

Compared to this approach the construction presented on the following pages is much more direct. Its "computational power" is more or less distributed among the monodromy behavior of several BISECT operations. Each of these angular bisectors contributes one bit of information to the "storage" of the device.

Our construction allows for variants and extensions that are not possible in the other approach. In particular, the results can be strengthened further to have only *one* free complex input variable. A streamlined variant of this result for the case of analytic continuation will be presented in [14].

6.1. **Another Gadget.**

Before starting the crucial construction of a binary counter we will introduce another gadget that simplifies this construction. We show that over the JMB construction set, we can intersect a line through the origin with the unit circle (note that there are no circles available in JMB). Let l be a line through the origin and $l_{\mathbb{R}}$ be the real axis. We consider the following GSP:

$$\begin{aligned} g &\leftarrow \text{BISECT}(l, l_{\mathbb{R}}) \\ q &\leftarrow \text{MEET}(l, \text{PARALLEL}(g, -1)) \end{aligned}$$

The output point q is one of the intersections of l and the unit circle. Which of the two intersections we get, depends on the choice of angular bisector. If the line l makes a half

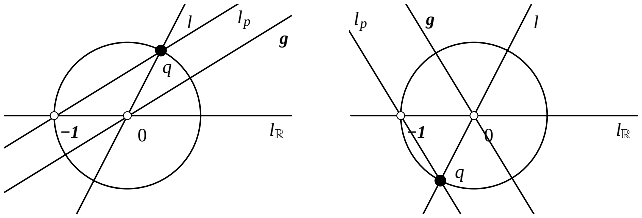


Fig. 12: Two instances of an INT_UNITCIRCLE(l) gadget.

turn (and by this comes back to its original position), the intersection moves continuously from one possibility to the other, see Fig. 12. We will encapsulate this construction within a (non-deterministic) macro INT_UNITCIRCLE(l) that produces one of the two intersections.

6.2. **A Binary Counter.** Our first sub-goal is to construct a binary counter, that drives the construction through an exponential number of different stages. For this we again start with a point z, which is given by

$$\begin{aligned} w &\leftarrow \text{ONCIRCLE}(-1,1,p), \\ z &\leftarrow w^4. \end{aligned}$$

As described in Sec. 5.1, while p moves on a straight vertical path from $a = 2$ to the point $b = 2+3i$, the point z makes one full counterclockwise cycle on the unit circle, starting and ending at 1. W.l.o.g. for our considerations we may assume that p is bound to lie on the line that connects a and b.

We will consider the point z directly as a driving input point that is bound to lie on the unit circle. We now have a look at the following functions:

$$\begin{array}{lll} z_1 \leftarrow & \mathrm{Re}(z^2) + z^{2^n} - 2 \\ z_2 \leftarrow & \mathrm{Re}(z^4) + z^{2^n} - 2 \\ z_3 \leftarrow & \mathrm{Re}(z^8) + z^{2^n} - 2 \\ \vdots & \vdots & \vdots \\ z_n \leftarrow & \mathrm{Re}(z^{2^n}) + z^{2^n} - 2 \end{array}$$

The Re$(\dots)$ operation (that extracts the real part of complex number) can be carried out geometrically by a projection to the real axis. All the $z_1,\dots,z_n$ can be constructed by a GSP whose total number of construction steps is linear in n. No angular bisectors are needed.

All the real parts of the functions $z_1,\dots,z_n$ are in the interval $[-2,0]$, since the numbers z^k all lie on the unit circle. While z moves along the unit circle the function z_k is zero exactly if z is a 2^k-th root of unity. Each of the functions z_k is real iff z is a 2^{n+1}-st root of unity. For $j = 0,\dots,2^n$ we set

$$a_j = e^{2i\pi(j/2^n)} \quad \text{and} \quad b_j = e^{2i\pi(j/2^n+1/2^{n+1})}.$$

We will refer to regions on the unit circle by the term *circular interval*. For two points a,b on the unit circle the open circular interval (a,b) is the (open) arc on the unit circle that arises by traveling counterclockwise from a to b.

The imaginary parts of the functions z_k only depend on the function z^{2^n}. Im(z_k) is positive in the circular intervals (a_j,b_j) and negative in the circular intervals (b_j,a_{j+1}) for $j = 0,\dots,2^n$. The largest purely real value that occurs for the function z_k is 0, the second largest real value is $\cos(2\pi/2^k)-1$. We set $\varepsilon = -\cos(2\pi/2^{n+1})+1$, a number that can be constructed geometrically by n successive BISECT operations of a right angle, followed by a projection.

Now let $l_{\mathbb{R}}$ be the real axis and we add for each $k = 1,\dots,n$ the following instructions to our GSP:

$$\begin{aligned} l_k &\leftarrow \text{BISECT}(\text{JOIN}(0, z_k + \varepsilon), l_{\mathbb{R}}) \\ q_k &\leftarrow \text{INT\_UNITCIRCLE}(l_k) \end{aligned}$$

First observe that the operation $\text{JOIN}(0,z_k+\varepsilon)$ is always admissible, since $z_k+\varepsilon$ is never 0. The line $\text{JOIN}(0,z_k+\varepsilon)$ is identical to $l_{\mathbb{R}}$ whenever z is either a_j or b_j for some $i \in \{0,\dots,2^n\}$. Thus at these places the line l_k can either be the real or the imaginary axis. Note that the only freely movable point in the whole construction so far is z (indirectly controlled by p). As a starting instance of the GSP we set $z = 1$. All lines $\text{JOIN}(0,z_k+\varepsilon)$ are then identical to $l_{\mathbb{R}}$. We get an admissible instance of the above operations by setting all $l_k = l_{\mathbb{R}}$ and setting all $q_k = 1$.

From this admissible starting instance A the behavior of the entire construction is determined (by analytic continuation) while z travels along the unit circle.

Lemma 6.3. *With all settings as above (starting at A) the values of the q_k are uniquely determined for all z (with $|z| = 1$). In particular, we have that for all $j \in \{0,\dots,2^n-1\}$ the value $z = b_j$ implies that*

$$q_k = \begin{cases} i & \text{for} \quad \sigma_{n-k} = 0, \\ -i & \text{for} \quad \sigma_{n-k} \neq 0. \end{cases}$$

Here $j = \sigma_0 2^0 + \sigma_1 2^1 + \sigma_2 2^2 + \cdots + \sigma_{n-1} 2^{n-1}$ with $\sigma_k \in \{0,1\}$ is the binary expansion.

PROOF. For the proof let us investigate what happens during a full counterclockwise cycle of the driving point z. At the beginning ($z = 1 = a_0$) all the l_k are by definition of A identical to the real axis and all q_k are by definition equal to 1. Furthermore, all z_k are positive (namely equal to ε). While z travels from a_0 to b_0 all z_k move through the upper halfplane to a negative value. Thus all lines $\text{JOIN}(0,z_k+\varepsilon)$ make a counterclockwise half turn. Consequently all l_k make a counterclockwise quarter turn and all q_k move to the value i as stated in the theorem.

Now let us investigate what happens when z moves from position b_j on the shortest possible path via a_{j+1} to position b_{j+1}. During such a move the position of q_k will make a half turn if and only if z_k makes a cycle around the origin. This in turn exactly happens if for $z = a_{j+1}$ the value of z_k is positive. However, this is only the case if a_{j+1} is a 2^k-th root of unity. This gives exactly the desired counting behavior. Finally after z has completed one full cycle, all elements are back to their initial positions. Hence no global monodromy occurs and the behavior is globally determined. □

The whole construction behaves like a binary counter. For each position $z = b_j$ all q_k are either i or $-i$. The positions exactly resemble the behavior of the binary expansion of j with i playing the role of the 0 and $-i$ playing the role of the 1.

6.3. **A Register.** The output of the counter we constructed so far will later on play the role of the x_k that occur in the quantified boolean formula

$$\forall x_1 \exists y_1 \forall x_2 \exists y_2 \dots \forall x_n \exists y_n f(x_1,y_1,x_2,y_2,\dots,x_n,y_n).$$

We now explain how to model the y_k. For this we construct a *register* with dependent points $r_1,\dots,r_n$. Whenever the driving point z is at a position b_j the r_k will either be i or $-i$. However, which of the two values will be taken depends on the position of certain free points $p_1,\dots,p_n$ during z being at a position a_j. We will call the points b_j *evaluation points* and call the points a_j *setting points*. For each $k \in \{1,\dots,n\}$ we add the following four lines to our GSP constructed so far:

$$\begin{aligned} p'_k &\leftarrow \text{ONINTERVAL}(0,1,p_k)\\ z'_k &\leftarrow p'_k - 1 + z_k + \varepsilon\\ l'_k &\leftarrow \text{BISECT}(\text{JOIN}(0,z'_k),l_{\mathbb{R}})\\ r_k &\leftarrow \text{INT\_UNITCIRCLE}(l_k)\end{aligned}$$

We extend our initial position A (remember there we had $z = 1$) first by setting $p_1 = p_2 = \dots = p_n = 2$. This forces the lines $\text{JOIN}(0,z'_k)$ to be $l_{\mathbb{R}}$. As we did for the counter we set all $l'_k = l_{\mathbb{R}}$ and all $r_k = 1$. The following Lemma summarizes the properties of the register.

Lemma 6.4. *For any admissible move of the free input points $p, p_1,\dots,p_n$ starting from instance A we have the following properties:*

(i) *Whenever z is at an evaluation point each of the r_k is either i or $-i$.*

(ii) *For each $k \in \{1,\dots,n\}$ and each $j \in \{0,\dots,2^k-1\}$ we have: Whenever z stays in the circular interval $I_{k,j} = (a_{j\cdot 2^{n-k}}, a_{(j+1)\cdot 2^{n-k}})$, all values of r_k when z is at an evaluation point in $I_{k,j}$ are identical.*

(iii) *Except conditions* (i) *and* (ii) *there are no other restrictions to the values of r_k when z is at an evaluation point.*

PROOF. The proof is very similar to the proof of Lemma 6.3. The initial situation of A ensures that condition (i) is satisfied. For each $j \in \{0,\dots,2^k-1\}$ the function z'_k is always negative within the entire circular interval $I_{k,j}$. Hence, as long as z does not leave this interval for all evaluation points the values of the r_k must be identical (since z'_k cannot cycle around the origin.) This proves (ii). The function z'_k *can* be positive whenever z takes one of the values $a_{j\cdot 2^{n-k}}$ with $j \in \{0,\dots,2^k-1\}$ (these are the 2^k-th roots of unity). Whether it actually *is* positive depends on the position of the free point p_k. Thus whenever z makes a transition between the intervals $I_{k,j}$ and $I_{k,j+1}$ (in either direction) we can (by moving p_k accordingly) control whether the r_k at z being at an evaluation point in $I_{k,j}$ and $I_{k,j+1}$ are identical for both intervals or not. This proves (iii). □

6.4. **Encoding the QBF.** Let us now step back and see what we have achieved so far. We have constructed a GSP together with an initial position A. The free points of the GSP are $p, p_1,,\dots,p_n$. We have output points $q_1,\dots,q_n$ for the counter and $r_1,\dots,r_n$ for the register. Whenever z is at an evaluation point all these output points are either i or $-i$. For $n = 3$ the "timing" of the whole construction is shown in Fig. 13. The horizontal axis shows the positions for z on the unit circle. Each of the oscillating curves roughly represents the values of the counting points q_1, q_2, q_3. The dots mark those positions that are relevant for possibly changing the values of the register points r_1, r_2, r_3. The bottom row represents the point pair (q_3, r_3). The middle row represents (q_2, r_2), and the top row represents (q_1, r_1). Between two of the positions marked with a dot the corresponding r-point always has the same value at the evaluation points. If for instance z passes a_0 in clockwise direction the position of p_1 determines which values r_1 can take at the evaluation points b_0, b_1, b_2, b_3. These values can only be changed if z passes once more one of the setting points a_0 or a_4.

We are now going to link a given QBF formula to our construction. For this consider the QBF

$$\forall x_1 \exists y_1 \forall x_2 \exists y_2 \dots \forall x_n \exists y_n\, f(x_1, y_1, x_2, y_2, \dots, x_n, y_n)$$

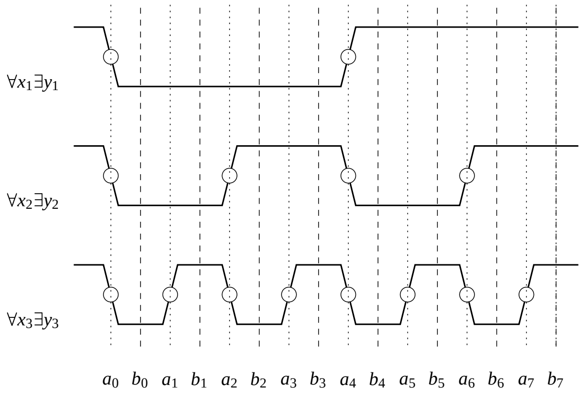

$\forall x_1 \exists y_1$
$\forall x_2 \exists y_2$
$\forall x_3 \exists y_3$
a_0 b_0 a_1 b_1 a_2 b_2 a_3 b_3 a_4 b_4 a_5 b_5 a_6 b_6 a_7 b_7

Fig. 14: The timing of the counter and the register

where we assume that f is given in conjunctive normal form. As in Sec. 4.1 we first translate the boolean formula f into a polynomial F_f by replacing each positive literal x_k by a real variable x_k and each negative literal $\neg x_k$ by $(1-x_k)$. We do so similarly for the y_k. Then each *and*-operation is replaced by a multiplication and each *or*-operation is replaced by an addition. The resulting polynomial is called $F_f(x_1,y_1,x_2,y_2,\ldots,x_n,y_n)$. Similar to Lemma 4.4 we obtain

Lemma 6.5. *We have $F_f(x_1,y_1,x_2,y_2,\ldots,x_n,y_n) = 0$ for some $(x_1,y_1,\ldots,x_n,y_n) \in [0,1]^{2n}$ if and only if all variables are either 0 or 1, and $f(b(x_1),b(y_1),\ldots,b(x_n),b(y_n)) = \text{TRUE}$ with $b(0) = \text{TRUE}$ and $b(1) = \text{FALSE}$. For all other choices of the variables in $[0,1]^{2n}$ the value of F_f is strictly positive. Furthermore, at each corner of the cube $[0,1]^{2n}$ the polynomial F_f evaluates to an integer number.*

PROOF. The proof is straightforward. It follows exactly the considerations in Sec. 4.1. □

We now (constructively) identify the values $-i$ and i (of the q_k and r_k) with the boolean values TRUE and FALSE, respectively by adding the statements

$$x_k \leftarrow \tfrac{1}{2}(iq_k+1),$$
$$y_k \leftarrow \tfrac{1}{2}(ir_k+1)$$

for $k = 1,\ldots,n$ to our GSP. Furthermore, we set with these newly constructed points

$$F \leftarrow F_f(x_1,y_1,x_2,y_2,\ldots,x_n,y_n).$$

The next lemma brings us close to the complexity result we are aiming for. Recall that the point p was w.l.o.g bound to the line connecting 2 and $2+3i$, and that while p moves straight from 2 to $2+3i$ the path of the point z is a full clockwise cycle on the unit circle. by

Lemma 6.6. *The QBF $\forall x_1\exists y_1\forall x_2\exists y_2\ldots\forall x_n\exists y_n f(x_1,y_1,x_2,y_2,\ldots,x_n,y_n)$ is true if and only if the following holds: In our GSP starting with instance A there is an admissible path that moves point p from 2 to $2+3i$ such that the value of F is 0 whenever z is at an evaluation point.*

PROOF. First assume that $\forall x_1\exists y_1\forall x_2\exists y_2\ldots\forall x_n\exists y_n f(x_1,y_1,x_2,y_2,\ldots,x_n,y_n)$ is true. We can skolemize the variables y_k of the $\exists$-quantors by introducing Skolem functions

$$s_1(x_1), s_2(x_1,x_2),\ldots,s_n(x_1,\ldots,x_n)$$

such that

$$f(x_1, s_1(x_1), x_2, s_2(x_1, x_2), \dots, x_n, s_n(x_1, \dots, x_n))$$

is a tautology. These Skolem functions are exactly the "strategy" of the two player game associated to the QBF, which tell player Y how to move. We can derive a path as desired in the theorem as follows. We first choose δ to be a sufficiently small positive number such that for $p = 2 - \delta i$ the point z lies in the circular interval (b_{2^n-1}, a_0) and move point p straight from 2 to $2 - \delta i$ while leaving the p_k unchanged.

After that we move point p straight from $2 - \delta i$ to $2 + 3i$. Whenever z passes one of the setting points we make sure that the positions of the points $p_1, \dots, p_n$ were adjusted such that for the forthcoming evaluation point the y_k take the values of the corresponding Skolem functions (this is possible by Lemma 6.4). The construction allows exactly enough freedom to possibly change the value of s_k whenever the variable x_k changes its value. By this choice and by Lemma 6.5 the derived point F is 0 at each evaluation point.

Conversely assume that we know how to move the points p_k such that point p can move from 2 to $2 + 3i$ in a way that whenever z is at an evaluation point the dependent point F is 0. We call such a path *correct*. We show how to reconstruct the Skolem functions from the situations at the evaluation points. One technical difficulty arises from the fact that it may happen that p changes its moving direction while traveling from 2 to $2 + 3i$ arbitrarily often. If this is the case the point z possibly meets several of the points a_j or b_j more than once.

Assume that the movement of $p = 2 + \phi(t)i$ is given by the function $\phi(t) : [0,1] \to \mathbb{R}$ with $\phi(0) = 0$ and $\phi(1) = 3$. The induced movement of z will be denoted by $z(t)$. Furthermore, we set $b(i) = \text{FALSE}$ and $b(-i) = \text{TRUE}$. Let us abbreviate TRUE by T and FALSE by F. We describe step by step how to derive the functions $s(x_1)$, $s(x_1, x_2), \dots$.

Since $z(t)$ (continuously) describes one full cycle there is an (open) interval $I_1^{\text{F}} \subset (0,1)$ such that $z(I_1^{\text{F}})$ equals the (open) circular interval $(a_0, a_{2^{n-1}})$. Lemma 6.5(ii) tells us, that within this interval the value of point r_1 must be the same for all evaluation points. Lemma 6.5(i) shows that this value r_1^{F} must be either i or $-i$. We set $s_1(\text{F}) = b(r_1^{\text{F}})$. Similarly there is an open interval $I_1^{\text{T}} \subset (0,1)$ such that $z(I_1^{\text{T}}) = (a_{2^{n-1}}, a_{2^n})$. Also there the value r_1^{T} at the evaluation points is uniquely either i or $-i$. We set $s_1(\text{T}) = b(r_1^{\text{T}})$.

We now proceed inductively. Within the interval I_1^{F} there are subintervals $I_2^{\text{F,F}}$ and $I_2^{\text{F,T}}$ such that $z(I_2^{\text{F,F}}) = (a_0, a_{2^{n-2}})$ and $z(I_2^{\text{F,F}}) = (a_{2^{n-2}}, a_{2^{n-1}})$. We let $s_2(\text{F},\text{F}) = b(r_2^{\text{F,F}})$ and $s_2(\text{F},\text{T}) = b(r_2^{\text{F,T}})$. Similarly we define the values $s_2(\text{T},\text{F})$ and $s_2(\text{T},\text{T})$ by considering suitable subintervals of I_1^{T}. The values of the Skolem functions $s_3, \dots, s_n$ are defined similarly each one by looking at a suitable subintervals of the intervals considered for the previous function. Now, by our initial assumption the value of F was 0 at each evaluation point. Lemma 6.5 thus ensures that

$$f(x_1, s_1(x_1), x_2, s_2(x_1, x_2), \dots, x_n, s_n(x_1, \dots, x_n))$$

is a tautology. Hence the original QBF was true. □

In the current construction the number of free points depends on the problem size. However, the character of the construction allows for an easy alternative that has just one free input point. We can strengthen Lemma 6.6 to the following version.

Lemma 6.7. *The GSP of Lemma 6.6 can be assumed to have only one free point p.*

PROOF. For this note that in order to get a correct path we have to set the points $p'_1, \dots, p'_n$ to suitable values in $\{0,1\}$ whenever z is at a setting point. We construct a second counter controlled by a free point p' on the segment from 2 to $2+3i$. This new counter is just an identical copy of the decvice described in Sec. 6.2. We connect the outputs $q'_1, \dots, q'_n$ of this new counter directly to the p'_k by setting (i.e. redefining)

$$p'_k \leftarrow \mathrm{Re}((iq'_k + 1)/2).$$

Now the position of the p'_k can be controlled by the position of p'. In particular every $0/1$-combination of the p'_k can be achieved by placing p' to a suitable position. This construction still behaves like the construction of Lemma 6.6 but it has only two input points p and p'. We now "rename" our input point p to p'', then we introduce a new free point p, and add the following instructions $p'' \leftarrow 2 + \mathrm{Im}(p)$ and $p' \leftarrow 2 + \mathrm{Re}(p-2)\cdot i$ to our GSP. This controls the points p' and p'' by the x and y parameter of just one single input point p. Still all necessary freedom that makes Lemma 6.6 work is maintained. □

6.5. **The Inequality Condition.** Our final task is now to transfer the "existence of a correct path"-property of Lemma 6.7 to a suitable geometric condition like the incircle condition in Thm. 6.1. For this we add the following statements to our GSP:

$$\begin{aligned} G &\leftarrow F + z^{2^n} + 1/2 \\ z' &\leftarrow \textsc{int\_unitcircle}(\textsc{join}(G,0)) \\ d &\leftarrow z_n - z' \end{aligned}$$

We resolve the nondeterminisms by setting $h = l_{\mathbb{R}}$ at our initial position A and $z' = 1$.

Lemma 6.8. *In our GSP starting with instance A consider an admissible path that moves point p from 2 to $2+3i$. For such a path the following conditions are equivalent:*

(i) *If z is at an evaluation point the value of F is 0.*
(ii) *We have $|d| < 2$ throughout the path.*

Proof. Since F is positive and stays on the real axis the value of G is real if and only if z^{2^n} is real. This is exactly the case if z is either at a setting point a_j (then $z^{2^n} = 1$) or at an evaluation point b_j (then $z^{2^n} = -1$). This implies that G is positive at all setting points. Remember that F assumes only integer values at the evaluation points. Hence G is negative at an evaluation point if and only if $F = 0$ at this evaluation point. Point z' is the image of this point G mapped by a central projection to the unit circle.

Now assume that the path has the property as claimed in (i). We prove that this path automatically satisfies (ii). In this path $z' = 1$ at each setting point, $z' = -1$ at each evaluation point. Furthermore, $\mathrm{sign}(\mathrm{Im}(z')) = \mathrm{sign}(\mathrm{Im}(z^{2^n}))$. Hence there is always a line through the origin that has z' and z^{2^n} on the same side and therefore we have $|d| < 2$ throughout the path.

Conversely, assume that there is no path that satisfies condition (i). This means that for every possible path along which p is moved from 2 to $2+3i$ the total winding number of z' with respect to the origin is less than 2^n. This is the case since z' can cross the negative half line only for all the evaluation points. On the other hand the total winding number of z^{2^n} with respect to the origin is 2^n. This implies that there is at least one position along the path where z and z^{2^n} are antipodals on the circle. At this position we have $d = 2$. □

Now we obtain Thm. 6.1 as a direct consequence of the PSPACE-hardness of QBF, Lemma 6.5 and Lemma 6.6. This finishes the proof of Thm. 6.1.

Remark 6.9. Without formal proof we mention a few possible alterations of Thm. 6.1.

- *Non-Strict inequalities:* The construction we gave really needed a *strict* inequality as additional obstruction (two points on the unit circle cannot be further apart than 2 units). It is easy to obtain the same result also with a non strict inequality. For instance one could introduce an additional BISECT operation of $\text{JOIN}(G,0)$ and $l_{\mathbb{R}}$ intersect the result with the unit circle and compare the resulting point with $z^{2^{n-1}}$.
- *Sidedness vs. incircle test:* One can turn an incircle condition $|d| < 1$ as used in Thm. 6.1 into a sidedness test of a point w.r.t. a line. One possibility for this is to construct the point
$$d' \leftarrow (\text{MEET}(\text{PERPENDICULAR}(\text{JOIN}(0,d),d),l_{\mathbb{R}}))^2.$$
The incircle condition then translates to $d' < 1$.
- *Admissibility of circle-line intersections:* We may also introduce the operation of intersecting the unit circle with a line. We may restrict the range of admissible situations to those where such an intersection properly exists. By this we can also express the inequality conditions that are needed in Thm. 6.1.
- *Restriction on total length of the path of p:* One can also perturb Thm. 6.1 in a way such that the inequality becomes a restriction on the total length of the path of p in a reachability problem. For this we first replace our assignment $p' \leftarrow 2 + \text{Re}(p-2)i$ by $p' \leftarrow 2 + N \cdot \text{Re}(p-2)i$ for a very large number N. This operation rescales the imaginary part of p such that the control of the p'_i does not really contribute significantly to the overall length of the path p takes. Then we take the $(n+1)$-times iterated angular bisector h_{n+1} of $\text{JOIN}(G,0)$ and $l_{\mathbb{R}}$ and ask the following reachability problem. "Is it possibly to move p from 2 to $2+3i$ such that h_{n+1} make a 90°-turn into its other alternative such that the total length of the path described by p does not exceed $3+\delta$ (for sufficiently small $\delta > 0$)?" The only way to do this is to move straight from 2 to $2+3i$ by passing every setting point at most once. At each evaluation point the function F must be 0 in order to end up with a rotation of the desired amount. This is by Lemma 6.5 equivalent to knowing the Skolem functions for the QBF. This variant is particularly important, since then no additional sidedness or incircle test is needed.
- *Games against external forces:* The last statement can reformulated also in another way. Redefine p' and p'' as free points again. Assume that an exterior force moves p'' from $2 - \delta i$ to $2 + 3i$. Can you simultaneously move the points p' such that h_{n+1} makes a 90°-turn? This is PSPACE-hard to decide.

7. Undecidable Problems

In this section we enlarge the set of our possible primitive operations. We add one non-deterministic operation that models the mechanical behavior of a *wheel* that rolls along a road. In principle wheels have the ability to transfer angles to distances. If the wheel was rotated for a certain angle it has traveled along the road for a certain distance. If we (as usual) denote angles modulo 2π this introduces a new kind of monodromy behavior to our context. For the same angle as input there is an infinity of possible output values. All these output lengths are integer multiples of the length that is generated by rotating the wheel once by 2π.

This new type of monodromy introduces a drastic change in the complexity behavior of the reachability problem. We will see that we can translate the solvability of Diophantine equations into a reachability problem for a construction involving several wheels. By the undecidability of Hibert's 10th problem this induces the undecidability for this reachability problem.

7.1. **Wheels.** Let us first formalize the concept of a wheel to fit into our setup of geometric straight line programs. The right algebraic function that models the behavior of wheels is the logarithm function applied to points on the unit circle. Our WHEEL-primitive will take a point $p = r \cdot e^{i\varphi} \neq 0$ and map it non-deterministically to a point on the real axis that represents the possible angles. We define the relation

$$\text{WHEEL} := \{(p,q) \mid q = \varphi/2\pi; q = r \cdot e^{i\varphi}\} \subset P \times P.$$

As usual we allow ourselves to write $q \leftarrow \text{WHEEL}(p)$ if $(p,q) \in \text{WHEEL}$. The operation is not admissible for $p = 0$. If $(p,q) \in \text{WHEEL}$ then we have also $(p, q+k) \in \text{WHEEL}$ for all $k \in \mathbb{Z}$. In particular we have $(1,k) \in \text{WHEEL}$ for all $k \in \mathbb{Z}$. If we start with an admissible instance $p = 1; q = 0$ of $q \leftarrow \text{WHEEL}(p)$ then we can continuously reach the situation $p = 1; q = k$ by letting p spin around the origin k times. In our picture of an actual mechanical wheel the operation WHEEL is designed to model the properties of a wheel of circumference 1 (and thus of irrational (!) radius $\frac{1}{2\pi}$). The resulting RIS is called JMBW.

7.2. **Diophantine Equations.** The following theorem states one version of the famous undecidability of Diophantine equations.

Theorem 7.1. *Let $N = 11$ and $f \in \mathbb{Z}[x_1, \ldots, x_N]$ be a polynomial. It is algorithmically undecidable whether f has a zero in $\mathbb{Z}^N$.*

The number N depends on the actual state of research in the area around Hilbert's 10th problem [31, 32]. We keep it fixed for the following considerations.

We will prove the following theorem by reduction to the above statement:

Theorem 7.2. *Let $\mathcal{P}$ be a GSP over the* JMBW *instruction set with at least N* WHEEL*-operations and two* BISECT*-operations. Let A and B be two admissible instances of $\mathcal{P}$. It is undecidable whether there is an admissible real path from A to B.*

PROOF. Step by step we will construct the translation from the polynomial in Thm. 7.1 to the GSP in Thm 7.2. We start with a free point p and a point z given by

$$z \quad \leftarrow \quad \text{ONCIRCLE}(-1, 1, p).$$

Then we add for $i = 1, \ldots, N$ the instructions:

$$\begin{array}{lcl} q_i & \leftarrow & \text{ONINTERVAL}(0, 2, p_i) \\ x_i & \leftarrow & \text{WHEEL}(q_i + z) \end{array}$$

Assume that in the instances A and B we have $p = p_1 = \ldots = p_N = (3,0)$ and thus $z = 1$, and $q_1 = \ldots = q_n = 2$. Any admissible instance with $z = 1$ that is reachable from A by an admissible path satisfies $x_i \in \mathbb{Z}$ for all $i = 1, \ldots, N$, since whenever $z = 1$ the $q_i + z$ are positive. Additionally, for every $(y_i, \ldots, y_N) \in \mathbb{Z}^N$ there is an admissible path starting at A and ending at a position with $z = 1$ and $x_i = y_i$ for all $i = 1, \ldots, N$. For this claim we only have to prove that each variable x_i can be changed by ±1 independently from the others. In order to obtain such an elementary change simply set $q_i = 0$ and $q_j = 2$ for $i \neq j$, and do a full clockwise or counterclockwise turn with point z.

Now let $f(x_1,\dots,x_N)$ be an instance of the polynomial used in Thm. 7.1. We finish our construction by adding the following statements to our GSP:

$$\begin{array}{rcl} F & \leftarrow & -(f(x_1,\dots,x_N))^2 - 3/2 \\ q_{N+1} & \leftarrow & \textsc{onInterval}(0,1,p_{N+1}) \\ l_0 & \leftarrow & \textsc{join}(q_{N+1}+F+z,0) \\ l_1 & \leftarrow & \textsc{bisect}(l_{\mathbb{R}},l_0) \\ l_2 & \leftarrow & \textsc{bisect}(l_{\mathbb{R}},l_1) \end{array}$$

For the starting instance A we assume that we have $p_{N+1} = -1$ and hence $q_{N+1} = 0$. This implies that $q_{N+1}+F$ is real. Thus in A we have $l_0 = l_{\mathbb{R}}$. For A we resolve the ambiguities of the BISECT-operations by setting $l_1 = l_2 = l_{\mathbb{R}}$. The only point in which instance B differs from A is that in B we set $l_2 = l_{i\mathbb{R}}$.

We now claim that it is undecidable whether instance B is reachable from instance A: Observe that the only way in which A can be transformed into B is that the point $q_{N+1} + F + z$ makes an odd number of cycles around the origin.

If there are $(y_i,\dots,y_N) \in \mathbb{Z}^N$ with $f(y_1,\dots,y_N) = 0$ we can achieve such a movement as follows. We set $q_{N+1} = 0$. By the procedure described above we follow a path that puts the x_i to the values of the y_i. Then we set all $q_1 = \dots q_N = 0$ and $q_{N+1} = 1$, do another full cycle with z, set $q_{N+1} = 0$ again, and reset to $x_i = 0$ again by a suitable movement. The resulting situation is exactly instance B.

Conversely assume that there is an admissible path from A to B. During this path we must have at least one position where $q_{N+1} + F + z$ is real and positive. This can only happen if $z = 1$, since $q_{N+1}+F$ is always real and by definition at most $-1/2$. However, if $z = 1$ the $f(x_1,\dots,x_N)$ must have an integer value. The only way to get $q_{N+1}+F+z > 0$ is to have $f(x_1,\dots,x_N) = 0$. Since all x_i were integral this means that a solution of the Diophantine equation exists. □

8. OPEN PROBLEMS

We end our considerations by stating at least some of the open problems in decision complexity of tracing and reachability.

Problem 8.1. *Determine upper bounds for the decision complexity in the various setups described in this paper.*

Problem 8.2. *Extend the* JMB *instruction set by a new non-deterministic operation that intersects a circle and a line. Furthermore, enlarge the setup such that also complex coordinates for points lines and circles are allowed. By this a circle and a line always have two or one intersections. What is the decision complexity of the reachability problem in this context?*

This problem is of fundamental importance, since if the intrinsic complexity would turn out not to be too big this might yield good algorithms for randomized theorem proving for ruler and compass theorems. The structure of this problem seems to be fundamentally different from the problems discussed in this paper. It is not unlikely that for this problem there are effective randomized methods. However, we are pessimistic about fast deterministic methods, since we can prove that it is at least as hard as zero testing for polynomials [13].

Closely related to the above problem is the following:

Problem 8.3. *Extend the* JMB *instruction set by a new non-deterministic operation that intersects a circle and a line. Furthermore, enlarge the setup such that also complex coordinates for points, lines, and circles are allowed. Let A be an instance of a construction and let B' be a partial instance that just defines the positions of the free elements. How difficult is it to complete B' to an instance B that is reachable from A?*

Our last problem forms another approach that may single out certain tracing problems to be more easy than others. Since it asks for a new concept, the formulation is kept a little vague on purpose.

Problem 8.4. *Define the "right" concept of* output sensitivity *for the tracing problem that allows statements like "if the elements move not too wildly we can trace them easily".*

Finally we ask for a slightly stronger version of Thm. 7.2 that gets rid of the constant π hidden in the WHEEL-operation.

Problem 8.5. *Redefine the relation* WHEEL *by*

$$\text{WHEEL} := \{(p,q) \mid q = \varphi; q = r \cdot e^{i\varphi}\} \subset P \times P.$$

Is Thm. 7.2 still valid in this setup?

References

[1] J. Culberson, *Sokoban is PSPACE-complete,* Proceedings in Informatics 4, Fun With Algorithms, E. Lodi, L. Pagli and N. Santoro Eds. pp 65-76, Carleton Scientific, Waterloo. 1999.

[2] M. J. Garey & D. S. Johnson, *Computers and Intractability,* W.H. Freeman and Company, New York (1979).

[3] J. R. Gilbert, T. Lengauer & R. E. Tarjan, *The pebbling problem is complete in polynomial space,* SIAM J. Comput., **9**, (1980), 513–524.

[4] H. Günzel, *The universal partition theorem for oriented matroids,* Discrete Comput. Geom., **19**, (1998), 521–551.

[5] Ch. M. Hoffmann, *Solid Modeling,* in: J.E. Goodman & J. O'Rourke. (eds.): *Handbook of Discrete and Computational Geometry*, Lecture Notes in Mathematics **1346**, CRC Press, Boca Raton, New York, 1997, 863–880.

[6] J. Hopcroft, D. Joseph & S. Whitesides, *Movement problems for 2-dimensional Linkages*, SIAM J. Comput., **13**, (1984), 610–629.

[7] J. Hopcroft, J.T. Schwarz & M. Sharir, *On the Complexity of Motion Planning for multiple Independent Objects; PSPACE-Hardness of the "Warehouseman's Problem",* Intern. J. Robotics Research **3**, (1984), 76–87.

[8] N. Jackiw, *The Geometer's Sketchpad*, Key Curriculum Press, Berkeley, 1991–1995.

[9] D. Jordan & M. Steiner, *Configuration Spaces of Mechanical Linkages,* Discrete Comput. Geom., **22**, (1999), 297–315.

[10] M Kapovich & J. Millson *On the moduli spaces of polygons in the Euclidean plane,* J. of Differential Geometry, **42** (1995), 133–164.

[11] U. Kortenkamp, *Foundations of Dynamic Geometry*, PhD-thesis, ETH Zürich, 1999, `http://www.inf.fu-berlin.de/∼kortenka/Papers/diss.pdf`.

[12] U. Kortenkamp & J. Richter-Gebert, *On the foundations of Dynamic Geometry*, in preparation.

[13] U. Kortenkamp & J. Richter-Gebert, *Decision complexity in Dynamic Geometry*, Proceedings of the ADG 2000, Springer Lecture Notes to, appear.

[14] U. Kortenkamp & J. Richter-Gebert, *The intrinsic complexity of analytic continuation*, in preparation.

[15] Jean-Marie Laborde and Franck Bellemain, *Cabri-Geometry II*, Texas Instruments, 1993–1998.

[16] N.E. Mnëv, *The universality theorems on the classification problem of configuration varieties and convex polytopes varieties,* in: Viro, O.Ya. (ed.): *Topology and Geometry – Rohlin Seminar,* Lecture Notes in Mathematics **1346**, Springer, Heidelberg 1988, 527–544.

[17] D. A. Plaisted, *Sparse Complex polynomials and polynomial reducibility,* Journal of Computers and System Sciences., **14**, (1977), 210–221.

[18] D. A. PLAISTED, *Some polynomial and integer divisibility problems are NP-hard,* SIAM J. Comput., **7**, (1978), 458–464.

[19] D. A. PLAISTED, *New NP-Hard and NP-Complete Polynomial and Integer Divisibility Problems,* Theoretical Computer Science, **31**, (1984), 125–138.

[20] J. RICHTER-GEBERT, *The Universality theorems for oriented matroids and polytopes,* Contemporary Mathematics **223**, (1999), 269–292.

[21] J. RICHTER-GEBERT, *Realization Spaces of Polytopes,* Lecture Notes in Mathematics **1643**, Springer, Heidelberg 1996.

[22] J. RICHTER-GEBERT & U. KORTENKAMP, *Cinderella - The interactive geometry software,* Springer 1999; see also `http://www.cinderella.de`.

[23] J. RICHTER-GEBERT & U. KORTENKAMP, *Cinderella - Die interaktive Geometriesoftware,* HEUREKA Klett, 2000.

[24] J. REIF, *Complexity of the movers' problem and generalizations,* Proc. 20th IEEE conf. on Foundations of Comp. Sci., Long beach, Calif.: IEEE Computer Society, 1979, 421–427.

[25] P. SHOR, *Stretchability of pseudolines is NP–hard,* in: *Applied Geometry and Discrete Mathematics – The Victor Klee Festschrift* (P. Gritzmann, B. Sturmfels, eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Amer. Math. Soc., Providence, RI, **4**, (1991), 531–554.

[26] M. SHUB & S. SMALE, *Complexity of Bezout's theorem I: Geometric Aspects,* J. Amer. Math. Soc., **6**, (1993), 459–501.

[27] M. SHUB & S. SMALE, *Complexity of Bezout's theorem II: Volumes and Probabilities,* in: *Computational Algebraic Geometry* (F. Eyssette and A. Galligo, eds.), Progress in Mathematics, Birkhauser, **109** (1993), 267–285.

[28] M. SHUB & S. SMALE, *Complexity of Bezout's theorem III: Condition number and packing,* J. Complexity, **9**, (1993), 4–14.

[29] M. SHUB & S. SMALE, *Complexity of Bezout's theorem IV: Probability of success,* SIAM Jour. of Numerical Analysis, **33**, (1996), 128–148.

[30] M. SHUB & S. SMALE, *Complexity of Bezout's theorem V: Polynomial Time,* Theoretical Computer Science, **133**, (1994), 141–164.

[31] Z. W. SUN, *Reduction of unknowns in Diophantine representations (English Summary,* Sci. Schina Ser A, **35**, (1992), 257–269.

[32] Z. W. SUN, *J.P. Jones' work on Hilbert's tenth problem and related topics,* Adv. in Math. (China), **22**, (1993), 312–331.

Jürgen Richter-Gebert
ETH Zürich
Inst. for Theoretical Computer Science
ETH Zentrum
CH-8092 Zürich
Switzerland
e-mail: `richter@inf.ethz.ch`

Ulrich H. Kortenkamp
FU Berlin
Institut für Informatik
Takustraße 9
D-14195 Berlin
Germany
e-mail: `kortenkamp@inf.fu-berlin.de`