Nonlinear Craig Interpolant Generation

Ting Gan[1], Bican Xia[2(✉)], Bai Xue[3,4], Naijun Zhan[3,4(✉)], and Liyun Dai[5]

[1] School of Computer Science, Wuhan University, Wuhan, China
ganting@whu.edu.cn
[2] LMAM & School of Mathematical Sciences, Peking University, Beijing, China
xbc@math.pku.edu.cn
[3] State Key Lab. of Comp. Sci., Institute of Software, CAS, Beijing, China
{xuebai,znj}@ios.ac.cn
[4] University of Chinese Academy of Sciences, Beijing, China
[5] RISE & School of Comp. and Inf. Sci., Southwest University, Chongqing, China
dailiyun@swu.edu.cn

Abstract. Craig interpolant generation for non-linear theory and its combination with other theories are still in infancy, although interpolation-based techniques have become popular in the verification of programs and hybrid systems where non-linear expressions are very common. In this paper, we first prove that a polynomial interpolant of the form $h(\mathbf{x}) > 0$ exists for two mutually contradictory polynomial formulas $\phi(\mathbf{x}, \mathbf{y})$ and $\psi(\mathbf{x}, \mathbf{z})$, with the form $f_1 \geq 0 \wedge \cdots \wedge f_n \geq 0$, where f_i are polynomials in $\mathbf{x}, \mathbf{y}$ or $\mathbf{x}, \mathbf{z}$, and the quadratic module generated by f_i is Archimedean. Then, we show that synthesizing such interpolant can be reduced to solving a semi-definite programming problem (SDP). In addition, we propose a verification approach to assure the validity of the synthesized interpolant and consequently avoid the unsoundness caused by numerical error in SDP solving. Besides, we discuss how to generalize our approach to general semi-algebraic formulas. Finally, as an applicaiton, we demonstrate how to apply our approach to invariant generation in program verification.

Keywords: Craig interpolant · Archimedean condition· Semi-definite programming · Program verification · Sum of squares

1 Introduction

Interpolation-based techniques have become popular in recent years because of their inherently modular and local reasoning, which can scale up existing formal verification techniques like theorem proving, model-checking, abstract interpretation, and so on, while the scalability is the bottleneck of these techniques. The study of interpolation was pioneered by Krajı́ček [20] and Pudlák [30] in connection with theorem proving, by McMillan in connection with model-checking [25], by Graf and Saïdi [14], Henzinger *et al.* [16] and McMillan [26] in connection with abstraction like CEGAR, by Wang *et al.* [17] in connection with machine-learning based program verification.

Craig interpolant generation plays a central role in interpolation-based techniques, and therefore has drawn increasing attention. In the literature, there are various efficient algorithms proposed for automatically synthesizing interpolants for decidable fragments of first-order logic, linear arithmetic, array logic, equality logic with uninterpreted functions (EUF), etc., and their combinations, and their use in verification, e.g., [26,16,37,18,33,19,6,27,33] and the references therein. Additionally, how to compare the strength of different interpolants is investigated in [9]. However, interpolant generation for non-linear theory and its combination with the aforementioned theories is still in infancy, although nonlinear polynomials inequalities are quite common in safety-critical software and embedded systems [39,38].

In [7], Dai *et al.* had a first try and gave an algorithm for generating interpolants for conjunctions of mutually contradictory nonlinear polynomial inequalities based on the existence of a witness guaranteed by Stengle's Positivstellensatz [36], which is computable using semi-definite programming (SDP). Their algorithm is incomplete in general but if all variables are bounded (called Archimedean condition), then it becomes complete. A major limitation of their work is that two mutually contradictory formulas ϕ and ψ must have the same set of variables. In [10], Gan *et al.* proposed an algorithm to generate interpolants for quadratic polynomial inequalities. The basic idea is based on the insight that for analyzing the solution space of concave quadratic polynomial inequalities, it suffices to linearize them by proving a generalization of Motzkin's transposition theorem for concave quadratic polynomial inequalities. Moreover, they also discussed how to generate interpolants for the combination of the theory of quadratic concave polynomial inequalities and *EUF* based on the hierarchical calculus proposed in [34] and used in [33]. Obviously, *quadratic concave* polynomial inequalities is a very restrictive class of polynomial formulas, although most of existing abstract domains fall within it as argued in [10]. Meanwhile, in [13], Gao and Zufferey presented an approach to extract interpolants for non-linear formulas possibly containing transcendental functions and differential equations from proofs of unsatisfiability generated by δ-decision procedure [12] based on interval constraint propagation (ICP) [1] by transforming proof traces from δ-complete decision procedures into interpolants that consist of Boolean combinations of linear constraints. Thus, their approach can only find the interpolants between two formulas whenever their conjunction is not δ-satisfiable. Similar idea was also reported in [21]. In [5], Chen *et al.* proposed an approach for synthesizing non-linear interpolants based on counterexample-guided and machine-learning, but it relies on quantifier elimination in order to guarantee the completeness and convergence, which gives rise to the low efficiency of their approach theoretically. In [35], Srikanth *et al.* presented an approach called *CAMPY* to exploit non-linear interpolant generation, which is achieved by abstracting non-linear formulas (possibly with non-polynomial expressions) to the theory of linear arithmetic with uninterpreted functions, i.e., EUFLIA, to prove and/or disprove if a given program satisfies a given property, that may contain nonlinear expressions.

Example 1. In order to compare the approach proposed in this paper and the ones aforementioned, consider

$$\begin{aligned}\phi = {} & -2xy^2 + x^2 - 3xz - y^2 - yz + z^2 - 1 \geq 0 \wedge 100 - x^2 - y^2 \geq 0 \wedge \\ & x^2z^2 + y^2z^2 - x^2 - y^2 + \frac{1}{6}(x^4 + 2x^2y^2 + y^4) - \frac{1}{120}(x^6 + y^6) - 4 \leq 0; \\ \psi = {} & 4(x-y)^4 + (x+y)^2 + w^2 - 133.097 \leq 0 \wedge 100(x+y)^2 - w^2(x-y)^4 - 3000 \geq 0.\end{aligned}$$

It can be checked that $\phi \wedge \psi \models \bot$.

Obviously, synthesizing interpolants for ϕ and ψ in this example is beyond the ability of the above approaches reported in [7,10]. Using the method in [13] implemented in dReal3 it would return "SAT" with $\delta = 0.001$, i.e., $\phi \wedge \psi$ is δ-satisfiable, and hence it cannot synthesize any interpolant using [12]'s approach with any precision greater than 0.001[6]. While, using our method, an interpolant $h > 0$ with degree 10 can be found as shown in Fig 1 [7]. Additionally, using the symbolic procedure REDUCE, it can be proved that $h > 0$ is indeed an interpolant of ϕ and ψ.

In this paper, we investigate this issue and consider how to synthesize an interpolant for two polynomial formulas $\phi(\mathbf{x}, \mathbf{y})$ and $\psi(\mathbf{x}, \mathbf{z})$ with $\phi(\mathbf{x}, \mathbf{y}) \wedge \psi(\mathbf{x}, \mathbf{z}) \models \bot$, where

$$\phi(\mathbf{x}, \mathbf{y}) : f_1(\mathbf{x}, \mathbf{y}) \geq 0 \wedge \cdots \wedge f_m(\mathbf{x}, \mathbf{y}) \geq 0,$$
$$\psi(\mathbf{x}, \mathbf{z}) : g_1(\mathbf{x}, \mathbf{z}) \geq 0 \wedge \cdots \wedge g_n(\mathbf{x}, \mathbf{z}) \geq 0,$$

$\mathbf{x} \in \mathbb{R}^r$, $\mathbf{y} \in \mathbb{R}^s$, $\mathbf{z} \in \mathbb{R}^t$ are variable vectors, $r, s, t \in \mathbb{N}$, and $f_1, \ldots, f_m, g_1, \ldots, g_n$ are polynomials. In addition, $\mathcal{M}_{\mathbf{x},\mathbf{y}}\{f_1(\mathbf{x}, \mathbf{y}), \ldots, f_m(\mathbf{x}, \mathbf{y})\}$ and $\mathcal{M}_{\mathbf{x},\mathbf{z}}\{g_1(\mathbf{x}, \mathbf{z}), \ldots, g_n(\mathbf{x}, \mathbf{z})\}$ are two Archimedean quadratic modules. Here we allow uncommon variables, that are not allowed in [7], and drop the constraint that polynomials must be concave and quadratic, which is assumed in [10]. The Archimedean condition amounts to that all the variables are bounded, which is reasonable in program verification, as only bounded numbers can be represented in computer in practice. We first prove that there exists a polynomial $h(\mathbf{x})$ such that $h(\mathbf{x}) = 0$ separates the state space of $\mathbf{x}$ defined by $\phi(\mathbf{x}, \mathbf{y})$ from the one defined by $\psi(\mathbf{x}, \mathbf{z})$ theoretically, and then propose an algorithm to compute such $h(\mathbf{x})$ based on SDP. Furthermore, we propose a verification approach to assure the validity of the synthesized interpolant and consequently avoid the unsoundness caused by numerical error in SDP solving. Finally, we also discuss how to extend our results to general semi-algebraic constraints.

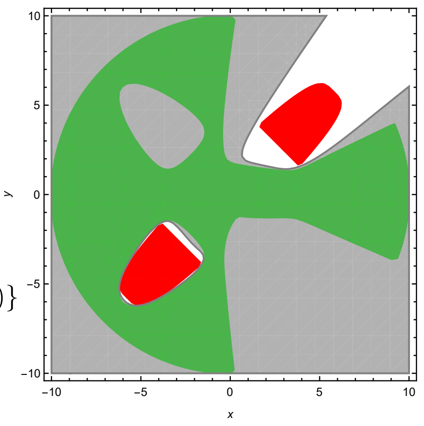

Fig. 1: Example 1. (Green region: the projection of $\phi(x, y, z)$ onto x and y; red region: the projection of $\psi(x, y, w)$ onto x and y; gray region plus the green region: the synthesized interpolant $\{(x, y) \mid h(x, y) > 0\}$.)

Another contribution of this paper is that as an application, we illustrate how to apply our approach to invariant generation in program verification by revising Lin *et al*'s framework proposed in [22] for invariant generation based on *weakest precondition*, *strongest postcondition* and *interpolation* by allowing to generate nonlinear invariants.

The paper is organized as follows. Some preliminaries and the problem of interest are introduced in Sect. 2. Sect. 3 shows the existence of an interpolant for two mutually contradictory polynomial formulas only containing conjunction, and Sect. 4 presents SDP-based methods to compute it. In Sect. 5, we discuss how to avoid unsoundness

[6] Alternatively, if we try the formula with the latest version of dReal4, it does not produce any output after 20 hours.

[7] The mathematical representation of h is given in the full version [11].

caused by numerical error in SDP. Sect. 6 extends our approach to general polynomial formulas. Sect. 7 demonstrates how to apply our approach to invariant generation in program verification. We conclude this paper in Sect. 8.

2 Preliminaries

In this section, we first give a brief introduction on some notions used throughout this paper and then describe the problem of interest.

2.1 Quadratic Module

$\mathbb{N}$, $\mathbb{Q}$ and $\mathbb{R}$ are the sets of integers, rational numbers and real numbers, respectively. $\mathbb{Q}[\mathbf{x}]$ and $\mathbb{R}[\mathbf{x}]$ denotes the polynomial ring over rational numbers and real numbers in $r \geq 1$ indeterminates $\mathbf{x} : (x_1, \ldots, x_r)$. We use $\mathbb{R}[\mathbf{x}]^2 := \{p^2 \mid p \in \mathbb{R}[\mathbf{x}]\}$ for the set of squares and $\sum \mathbb{R}[\mathbf{x}]^2$ for the set of sums of squares of polynomials in $\mathbf{x}$. Vectors are denoted by boldface letters. $\bot$ and $\top$ stand for **false** and **true**, respectively.

Definition 1 (Quadratic Module [24]). *A subset $\mathcal{M}$ of $\mathbb{R}[\mathbf{x}]$ is called a* quadratic module *if it contains 1 and is closed under addition and multiplication with squares, i.e., $1 \in \mathcal{M}, \mathcal{M} + \mathcal{M} \subseteq \mathcal{M}$, and $p^2\mathcal{M} \subseteq \mathcal{M}$ for all $p \in \mathbb{R}[\mathbf{x}]$.*

Let $\overline{p} := \{p_1, \ldots, p_s\}$ be a finite subset of $\mathbb{R}[\mathbf{x}]$, the quadratic module $\mathcal{M}_{\mathbf{x}}(\overline{p})$ or simply $\mathcal{M}(\overline{p})$ generated by $\overline{p}$ (i.e. the smallest quadratic module containing all p_is) is $\mathcal{M}_{\mathbf{x}}(\overline{p}) = \{\sum_{i=0}^{s} \delta_i p_i \mid \delta_i \in \sum \mathbb{R}[\mathbf{x}]^2\}$, where $p_0 = 1$.

Archimedean condition plays a key role in the study of polynomial optimization.

Definition 2 (Archimedean). *Let $\mathcal{M}$ be a quadratic module of $\mathbb{R}[\mathbf{x}]$. $\mathcal{M}$ is said to be* Archimedean *if there exists some $a > 0$ such that $a - \sum_{i=1}^{r} x_i^2 \in \mathcal{M}$.*

2.2 Problem Description

Craig showed that given two formulas ϕ and ψ in a first-order theory $\mathcal{T}$, if $\phi \models \psi$, then there always exists an *interpolant* I over the common symbols of ϕ and ψ s.t. $\phi \models I$ and $I \models \psi$. In the verification literature, this terminology has been abused following [26], where a *reverse interpolant* (coined by Kovács and Voronkov in [19]) I over the common symbols of ϕ and ψ is defined by

Definition 3 (Interpolant). *Given two formulas ϕ and ψ in a theory $\mathcal{T}$ s.t. $\phi \wedge \psi \models_{\mathcal{T}} \bot$, a formula I is an* interpolant *of ϕ and ψ if (i) $\phi \models_{\mathcal{T}} I$; (ii) $I \wedge \psi \models \bot$; and (iii) I only contains common symbols and free variables shared by ϕ and ψ.*

Definition 4. *A basic semi-algebraic set $\{\mathbf{x} \in \mathbb{R}^n \mid \bigwedge_{i=1}^{s} p_i(\mathbf{x}) \geq 0\}$ is called a set of the* Archimedean form *if $\mathcal{M}_{\mathbf{x}}\{p_1(\mathbf{x}), \ldots, p_s(\mathbf{x})\}$ is Archimedean, where $p_i(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, $i = 1, \ldots, s$.*

The interpolant synthesis problem of interest in this paper is given in **Problem** 1.

Problem 1. Let $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$ be two polynomial formulas of the form

$$\phi(\mathbf{x},\mathbf{y}) : f_1(\mathbf{x},\mathbf{y}) \geq 0 \wedge \cdots \wedge f_m(\mathbf{x},\mathbf{y}) \geq 0,$$
$$\psi(\mathbf{x},\mathbf{z}) : g_1(\mathbf{x},\mathbf{z}) \geq 0 \wedge \cdots \wedge g_n(\mathbf{x},\mathbf{z}) \geq 0,$$

where, $\mathbf{x} \in \mathbb{R}^r$, $\mathbf{y} \in \mathbb{R}^s$, $\mathbf{z} \in \mathbb{R}^t$ are variable vectors, $r, s, t \in \mathbb{N}$, and $f_1, \ldots, f_m, g_1, \ldots, g_n$ are polynomials in the corresponding variables. Suppose $\phi \wedge \psi \models \bot$, and $\{(\mathbf{x},\mathbf{y}) \mid \phi(\mathbf{x},\mathbf{y})\}$ and $\{(\mathbf{x},\mathbf{z}) \mid \psi(\mathbf{x},\mathbf{z})\}$ are semi-algebraic sets of the Archimedean form. Find a polynomial $h(\mathbf{x})$ such that $h(\mathbf{x}) > 0$ is an interpolant for ϕ and ψ.

3 Existence of Interpolants

The basic idea and steps of proving the existence of interpolants are as follows: Because an interpolant of ϕ and ψ contains only the common symbols in ϕ and ψ, it is natural to consider the projections of the sets defined by ϕ and ψ on $\mathbf{x}$, i.e. $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})) \hat{=} \{\mathbf{x} \mid \exists \mathbf{y}.\, \phi(\mathbf{x},\mathbf{y})\}$ and $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z})) \hat{=} \{\mathbf{x} \mid \exists \mathbf{z}.\, \psi(\mathbf{x},\mathbf{z})\}$, which are obviously disjoint. We therefore prove that, if $h(\mathbf{x}) = 0$ separates $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$, then $h(\mathbf{x})$ solves **Problem** 1 (see Proposition 1). Thus, we only need to prove the existence of such $h(\mathbf{x})$ through the following steps: First, we prove that $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$ are compact semi-algebraic sets which are unions of finitely many basic closed semi-algebraic sets (see Lemma 1). Second, using Putinar's Positivstellensatz, we prove that, for two disjoint basic closed semi-algebraic sets S_1 and S_2 of the Archimedean form, there exists a polynomial $h_1(\mathbf{x})$ such that $h_1(\mathbf{x}) = 0$ separates S_1 and S_2 (see Lemma 2). This result is then extended to the case that S_2 is a finite union of basic closed semi-algebraic sets (see Lemma 3). Finally, by generalizing Lemma 3 to the case that two compact semi-algebraic sets both are unions of finitely many basic closed semi-algebraic sets and together with Proposition 1, we prove the existence of interpolant in Theorem 2 and Corollary 1.

Proposition 1. *If $h(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ satisfies the following constraints*

$$\forall \mathbf{x} \in P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})).h(\mathbf{x}) > 0 \quad \textit{and} \quad \forall \mathbf{x} \in P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z})).h(\mathbf{x}) < 0, \tag{1}$$

*then $h(\mathbf{x}) > 0$ is an interpolant for $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$, where $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$ are defined as in **Problem** 1.*

Proof. According to Definition 3, it is enough to prove that $\phi(\mathbf{x},\mathbf{y}) \models h(\mathbf{x}) > 0$ and $\psi(\mathbf{x},\mathbf{z}) \models h(\mathbf{x}) \leq 0$.

Since any $(\mathbf{x}_0,\mathbf{y}_0)$ satisfying $\phi(\mathbf{x},\mathbf{y})$ must imply $\mathbf{x}_0 \in P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$, it follows that $h(\mathbf{x}_0) > 0$ from (1) and $\phi(\mathbf{x},\mathbf{y}) \models h(\mathbf{x}) > 0$. Similarly, we can prove $\psi(\mathbf{x},\mathbf{z}) \models h(\mathbf{x}) < 0$, implying that $\psi(\mathbf{x},\mathbf{z}) \models h(\mathbf{x}) \leq 0$. Therefore, $h(\mathbf{x}) > 0$ is an interpolant for $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$. □

In order to synthesize such $h(\mathbf{x})$ in Proposition 1, we first dig deeper into the two sets $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$. As shown later, i.e. in Lemma 1 , we will find that these two sets are compact semi-algebraic sets of the form $\{\mathbf{x} \mid \bigvee_{i=1}^{c} \bigwedge_{j=1}^{J_i} \alpha_{i,j}(\mathbf{x}) \geq$

$0\}$. Before this lemma, we introduce Finiteness theorem pertinent to a *basic closed semi-algebraic subset* of $\mathbb{R}^n$, which will be used in the proof of Lemma 1, where a basic closed semi-algebraic subset of $\mathbb{R}^n$ is a set of the form $\{\mathbf{x} \in \mathbb{R}^n \mid \alpha_1(\mathbf{x}) \geq 0, \ldots, \alpha_k(\mathbf{x}) \geq 0\}$ with $\alpha_1, \ldots, \alpha_k \in \mathbb{R}[\mathbf{x}]$.

Theorem 1 (Finiteness Theorem, Theorem 2.7.2 in [3]). *Let $A \subset \mathbb{R}^n$ be a closed semi-algebraic set. Then A is a finite union of basic closed semi-algebraic sets.*

Lemma 1. *The set $P_{\mathbf{x}}(\phi(\mathbf{x}, \mathbf{y}))$ is compact semi-algebraic set of the following form*

$$P_{\mathbf{x}}(\phi(\mathbf{x}, \mathbf{y})) := \{\mathbf{x} \mid \bigvee_{i=1}^{c} \bigwedge_{j=1}^{J_i} \alpha_{i,j}(\mathbf{x}) \geq 0\},$$

where $\alpha_{i,j}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, $i = 1, \ldots, c$, $j = 1, \ldots, J_i$. The same claim applies to the set $P_{\mathbf{x}}(\psi(\mathbf{x}, \mathbf{z}))$ as well.

Proof. For the sake of simplicity, we denote $\{(\mathbf{x}, \mathbf{y}) \mid \phi(\mathbf{x}, \mathbf{y})\}$ and $P_{\mathbf{x}}(\phi(\mathbf{x}, \mathbf{y}))$ by S and $\pi(S)$, respectively.

Because S is a compact set and π is a continuous map that maps compact set to compact set, $\pi(S)$, which is the image of a compact set under a continuous map, is compact. Moreover, as S is a semi-algebraic set and the projection of a semi-algebraic set is also a semi-algebraic set by Tarski-Seidenberg theorem [2], this implies that $\pi(S)$ is a semi-algebraic set. Thus, $\pi(S)$ is a compact semi-algebraic set.

Since $\pi(S)$ is a compact semi-algebraic set, and also a closed semi-algebraic set, we have that $\pi(\mathrm{S})$ is a finite union of basic closed semi-algebraic sets from Theorem 1. Hence, there exist a series of polynomials $\alpha_{1,1}(\mathbf{x})$, ..., $\alpha_{1,J_1}(\mathbf{x})$, ..., $\alpha_{c,1}(\mathbf{x})$, ..., $\alpha_{c,J_c}(\mathbf{x})$ such that $\pi(\mathrm{S}) = \bigcup_{i=1}^{c}\{\mathbf{x} \mid \bigwedge_{j=1}^{J_i} \alpha_{i,j}(\mathbf{x}) \geq 0\} = \{\mathbf{x} \mid \bigvee_{i=1}^{c} \bigwedge_{j=1}^{J_i} \alpha_{i,j}(\mathbf{x}) \geq 0\}$. This concludes this lemma. □

After knowing the structure of $P_{\mathbf{x}}(\phi(\mathbf{x}, \mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x}, \mathbf{z}))$ being a union of some basic semialgebraic sets as illustrated in Lemma 1, we next prove the existence of $h(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ satisfying (1), as formally stated in Theorem 2.

Theorem 2. *Suppose that $\phi(\mathbf{x}, \mathbf{y})$ and $\psi(\mathbf{x}, \mathbf{z})$ are defined as in* ***Problem*** *1, then there exists a polynomial $h(\mathbf{x})$ satisfying* (1).

As pointed out by an anonymous reviewer that Theorem 2 can be obtained by some properties of the ring of Nash functions proved in [29]. In what follows, we give a simpler and more intuitive proof. To the end, it requires some preliminaries first. The main tool in our proof is Putinar's Positivstellensatz, as formulated in Theorem 3.

Theorem 3 (Putinar's Positivstellensatz [31]). *Let $p_1, \ldots, p_k \in \mathbb{R}[\mathbf{x}]$ and $S_1 = \{\mathbf{x} \mid p_1(\mathbf{x}) \geq 0, \ldots, p_k(\mathbf{x}) \geq 0\}$. Assume that the quadratic module $\mathcal{M}(p_1, \ldots, p_k)$ is Archimedean. For $q \in \mathbb{R}[\mathbf{x}]$, if $q > 0$ on S_1 then $q \in \mathcal{M}(p_1, \ldots, p_k)$.*

With Putinar's Positivstellensatz we can draw a conclusion that there exists a polynomial such that its zero level set[8] separates two compact semi-algebraic sets of the Archimedean form, as claimed in Lemmas 2 and 3.

[8] The zero level set of an n-variate polynomial $h(\mathbf{x})$ is defined as $\{\mathbf{x} \in \mathbb{R}^n \mid h(\mathbf{x}) = 0\}$.

Lemma 2. *Let* $S_1 = \{\mathbf{x} \mid p_1(\mathbf{x}) \geq 0, \dots, p_J(\mathbf{x}) \geq 0\}$, $S_2 = \{\mathbf{x} \mid q_1(\mathbf{x}) \geq 0, \dots, q_K(\mathbf{x}) \geq 0\}$ *be semi-algebraic sets of the Archimedean form and* $S_1 \cap S_2 = \emptyset$, *then there exists a polynomial* $h_1(\mathbf{x})$ *such that*

$$\forall \mathbf{x} \in S_1.\, h_1(\mathbf{x}) > 0, \quad \forall \mathbf{x} \in S_2.\, h_1(\mathbf{x}) < 0. \tag{2}$$

Proof. Since $S_1 \cap S_2 = \emptyset$, it follows

$$p_2 \geq 0 \wedge \cdots \wedge p_J \geq 0 \wedge q_1 \geq 0 \wedge \cdots \wedge q_K \geq 0 \models -p_1 > 0.$$

Let $S_3 = \{\mathbf{x} \mid p_2 \geq 0 \wedge \cdots \wedge p_J \geq 0 \wedge q_1 \geq 0 \wedge \cdots \wedge q_K \geq 0\}$, then $-p_1 > 0$ on S_3. Since S_1 and S_2 are semi-algebraic sets of the Archimedean form, it follows $\mathcal{M}_{\mathbf{x}}(p_2(\mathbf{x}), \dots, p_J(\mathbf{x}), q_1(\mathbf{x}), \dots, q_K(\mathbf{x}))$ is also Archimedean. Hence, S_3 is compact. From $-p_1 > 0$ on S_3, we further have that there exists some $u_1 \in \sum \mathbb{R}[\mathbf{x}]^2$ such that $-u_1 p_1 - 1 > 0$ on S_3. Using Theorem 3, we have that

$$-u_1 p_1 - 1 \in \mathcal{M}_{\mathbf{x}}(p_2(\mathbf{x}), \dots, p_J(\mathbf{x}), q_1(\mathbf{x}), \dots, q_K(\mathbf{x})),$$

implying that there exists a set of sums of squares polynomials $u_2, \dots, u_J$ and v_0,v_1, $\dots$, $v_K \in \mathbb{R}[\mathbf{x}]$, such that

$$-u_1 p_1 - 1 \equiv u_2 p_2 + \cdots + u_J p_J + v_0 + v_1 q_1 + \cdots + v_K q_K.$$

Let $h_1 = \frac{1}{2} + u_1 p_1 + \cdots + u_J p_J$, i.e., $-h_1 = \frac{1}{2} + v_0 + v_1 q_1 + \cdots + v_K q_K$. It is easy to check that (2) holds. □

Lemma 3 generalizes the result of Lemma 2 to more general compact semi-algebraic sets of the Archimedean form, which is the union of multiple basic semi-algebraic sets.

Lemma 3. *Assume* $S_0 = \{\mathbf{x} \mid p_1(\mathbf{x}) \geq 0, \dots, p_J(\mathbf{x}) \geq 0\}$ *and* $S_i = \{\mathbf{x} \mid q_{i,1}(\mathbf{x}) \geq 0, \dots, q_{i,K_i}(\mathbf{x}) \geq 0\}$, $i = 1, \dots, b$, *are semi-algebraic sets of the Archimedean form, and* $S_0 \cap \bigcup_{i=1}^{b} S_i = \emptyset$, *then there exists a polynomial* $h_0(\mathbf{x})$ *such that*

$$\forall \mathbf{x} \in S_0.\, h_0(\mathbf{x}) > 0, \quad \forall \mathbf{x} \in \bigcup_{i=1}^{b} S_i.\, h_0(\mathbf{x}) < 0. \tag{3}$$

In order to prove this lemma, we prove the following lemma first.

Lemma 4. *Let* $c, d \in \mathbb{R}$ *with* $0 < c < d$ *and* $U_0 = [c, d]^r$. *There exists a polynomial* $\hat{h}(\mathbf{x})$ *such that*

$$\mathbf{x} \in U_0 \models \hat{h}(\mathbf{x}) > 0 \models \bigwedge_{i=1}^{r} x_i > 0, \tag{4}$$

where $\mathbf{x} = (x_1, \dots, x_r)$.

Proof. We show that there exists $k \in \mathbb{N}$ such that $\hat{h}(\mathbf{x}) = (\frac{d}{2})^{2k} - (x_1 - \frac{c+d}{2})^{2k} - \cdots - (x_r - \frac{c+d}{2})^{2k}$ satisfies (4). It is evident that $\hat{h}(\mathbf{x}) > 0 \models \bigwedge_{i=1}^{r} x_i > 0$ holds. In the following we just need to verify that $\bigwedge_{i=1}^{r} c \le x_i \le d \models \hat{h}(\mathbf{x}) > 0$ holds. Since $c \le x_i \le d$, we have $(x_i - \frac{c+d}{2})^{2k} \le (\frac{d-c}{2})^{2k}$ and $(\frac{d}{2})^{2k} - \sum_{i=1}^{r}(x_i - \frac{c+d}{2})^{2k} \ge (\frac{d}{2})^{2k} - r(\frac{d-c}{2})^{2k}$. Obviously, if an integer k satisfies $(\frac{d}{d-c})^{2k} > r$, then $(\frac{d}{2})^{2k} - \sum_{i=1}^{r}(x_i - \frac{c+d}{2})^{2k} > 0$. The existence of such k satisfying $(\frac{d}{d-c})^{2k} > r$ is assured by $\frac{d}{d-c} > 1$. □

Now we give a proof for Lemma 3 as follows.

Proof (of Lemma 3). For any i with $1 \le i \le b$, according to Lemma 2, there exists a polynomial $h_i \in \mathbb{R}[\mathbf{x}]$, satisfying $\forall \mathbf{x} \in S_0.\ h_i(\mathbf{x}) > 0$ and $\forall \mathbf{x} \in S_i.\ h_i(\mathbf{x}) < 0$.

Next, we construct $h_0(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ from $h_1(\mathbf{x}), \ldots, h_b(\mathbf{x})$. Since S_0 is a semi-algebraic set of the Archimedean form, S_0 is compact and thus $h_i(\mathbf{x})$ has minimum value and maximum value on S_0, denoted by c_i and d_i respectively. Let $c = \min(c_1, \ldots, c_b)$ and $d = \max(d_1, \ldots, d_b)$. Clearly, $0 < c < d$.

From Lemma 4 there must exist a polynomial $\hat{h}(w_1, \ldots, w_b)$ such that

$$\bigwedge_{i=1}^{b} c \le w_i \le d \models \hat{h}(w_1, \ldots, w_b) > 0, \tag{5}$$

$$\hat{h}(w_1, \ldots, w_b) > 0 \models \bigwedge_{i=1}^{b} w_i > 0. \tag{6}$$

Let $h_0'(\mathbf{x}) = \hat{h}(h_1(\mathbf{x}), \ldots, h_b(\mathbf{x}))$. Obviously, $h_0'(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$. We next prove that $h_0'(\mathbf{x})$ satisfies (3) in Lemma 3.

For all $\mathbf{x}_0 \in S_0$, $c \le h_i(\mathbf{x}_0) \le d$, $i = 1, \ldots, b$, $h_0'(\mathbf{x}_0) = \hat{h}(h_1(\mathbf{x}_0), \ldots, h_b(\mathbf{x}_0))$ > 0 by (5). Therefore, the first constraint in (3), i.e. $\forall \mathbf{x}_0 \in S_0. h_0(\mathbf{x}_0) > 0$, holds.

For any $\mathbf{x}_0 \in \bigcup_{i=1}^{b} S_i$, there must exist some i such that $\mathbf{x}_0 \in S_i$, implying that $h_i(\mathbf{x}_0) < 0$. By (6) we have $h_0'(\mathbf{x}_0) = \hat{h}(h_1(\mathbf{x}_0), \ldots, h_b(\mathbf{x}_0)) \le 0$.

Thus, we obtain the conclusion that there exists a polynomial $h_0'(\mathbf{x})$ such that $\forall \mathbf{x} \in S_0.\ h_0'(\mathbf{x}) > 0$, and $\forall \mathbf{x} \in \bigcup_{i=1}^{b} S_i.\ h_0'(\mathbf{x}) \le 0$. Also, since S_0 is a compact set, and $h_0'(\mathbf{x}) > 0$ on S_0, there must exist some positive number $\epsilon > 0$ such that $h_0'(\mathbf{x}) - \epsilon > 0$ over S_0. Then $h_0'(\mathbf{x}) - \epsilon < 0$ on $\bigcup_{i=1}^{b} S_i$. Therefore, setting $h_0(\mathbf{x}) := h_0'(\mathbf{x}) - \epsilon$, Lemma 3 is proved. □

In Lemma 3 we proved that there exists a polynomial $h(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ such that its zero level set is a barrier between two semi-algebraic sets of the Archimedean form, of which one set is a union of finitely many basic semi-algebraic sets. In the following we will give a formal proof of Theorem 2, which is a generalization of Lemma 3.

Proof (of Theorem 2). According to Lemma 1 we have that $P_{\mathbf{x}}(\phi(\mathbf{x}, \mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x}, \mathbf{z}))$ are compact sets, and there respectively exists a set of polynomials $p_{i,j}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, $i = 1, \ldots, a$, $j = 1, \ldots, J_i$, and $q_{l,k}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, $l = 1, \ldots, b$, $k = 1, \ldots, K_i$, such that

$$P_{\mathbf{x}}(\phi(\mathbf{x}, \mathbf{y})) = \{\mathbf{x} \mid \bigvee_{i=1}^{a} \bigwedge_{j=1}^{J_i} p_{i,j}(\mathbf{x}) \ge 0\}, \quad P_{\mathbf{x}}(\psi(\mathbf{x}, \mathbf{z})) = \{\mathbf{x} \mid \bigvee_{l=1}^{b} \bigwedge_{k=1}^{K_l} q_{l,k}(\mathbf{x}) \ge 0\}.$$

Since $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$ are compact sets, there exists a positive $N \in \mathbb{R}$ such that $f = N - \sum_{i=1}^{r} x_i^2 \geq 0$ over $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$. For each $i = 1,\ldots,a$ and each $l = 1,\ldots,b$, set $p_{i,0} = q_{l,0} = f$. Denote $\{\mathbf{x} \mid \bigvee_{i=1}^{a} \bigwedge_{j=0}^{J_i} p_{i,j}(\mathbf{x}) \geq 0\} = \bigcup_{i=1}^{a} \{\mathbf{x} \mid \bigwedge_{j=0}^{J_i} p_{i,j}(\mathbf{x}) \geq 0\}$ by P_1 and $\{\mathbf{x} \mid \bigvee_{l=1}^{b} \bigwedge_{k=0}^{K_l} q_{l,k}(\mathbf{x}) \geq 0\} = \bigcup_{l=1}^{b} \{\mathbf{x} \mid \bigwedge_{k=0}^{K_l} q_{l,k}(\mathbf{x}) \geq 0\}$ by P_2. It is easy to see that $P_1 = P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})$, $P_2 = P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$.

Since $\phi \wedge \psi \models \perp$, there does not exist $(\mathbf{x},\mathbf{y},\mathbf{z}) \in \mathbb{R}^{r+s+t}$ that satisfies $\phi \wedge \psi$, implying that $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})) \cap P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z})) = \emptyset$ and thus $P_1 \cap P_2 = \emptyset$. Also, since $\{\mathbf{x} \mid \bigwedge_{j=0}^{J_{i_1}} p_{i_1,j}(\mathbf{x}) \geq 0\} \subseteq P_1$, for each $i_1 = 1,\ldots,a$, $\{\mathbf{x} \mid \bigwedge_{j=0}^{J_{i_1}} p_{i_1,j}(\mathbf{x}) \geq 0\} \cap P_2 = \emptyset$ holds. By Lemma 3 there exists $h_{i_1}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ such that

$$\forall \mathbf{x} \in \{\mathbf{x} \mid \bigwedge_{j=0}^{J_{i_1}} p_{i_1,j}(\mathbf{x}) \geq 0\}.h_{i_1}(\mathbf{x}) > 0, \quad \forall \mathbf{x} \in P_2.h_{i_1}(\mathbf{x}) < 0.$$

Let $S' = \{\mathbf{x} \mid -h_1(\mathbf{x}) \geq 0,\ldots,-h_a(\mathbf{x}) \geq 0, N - \sum_{i=1}^{r} x_i^2 \geq 0\}$. Obviously, S' is a semialgebraic set of the Archimedean form, $P_2 \subset S'$ and $P_1 \cap S' = \emptyset$. Therefore, according to Lemma 2, there exists a polynomial $\overline{h}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ such that $\forall \mathbf{x} \in S'.\ \overline{h}(\mathbf{x}) > 0$ and $\forall \mathbf{x} \in P_1.\ \overline{h}(\mathbf{x}) < 0$. Let $h(\mathbf{x}) = -\overline{h}(\mathbf{x})$, then we have $\forall \mathbf{x} \in P_1.\ h(\mathbf{x}) > 0$ and $\forall \mathbf{x} \in P_2.\ h(\mathbf{x}) < 0$, implying that $\forall \mathbf{x} \in P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})).h(\mathbf{x}) > 0$ and $\forall \mathbf{x} \in P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z})).h(\mathbf{x}) < 0$. Thus, this completes the proof of Theorem 2. □

Consequently, we immediately have the following conclusion.

Corollary 1. *Let $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$ be defined as in **Problem** 1. There must exist a polynomial $h(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ such that $h(\mathbf{x}) > 0$ is an interpolant for ϕ and ψ.*

Actually, since $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$ both are compact set by Lemma 1, and $h(\mathbf{x}) > 0$ on $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $h(\mathbf{x}) < 0$ on $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$, we can obtain $h'(\mathbf{x})$ by giving a small perturbation to the coefficients of $h(\mathbf{x})$ such that $h'(\mathbf{x})$ has the property of $h(\mathbf{x})$. Hence, there should exist a $h(\mathbf{x}) \in \mathbb{Q}[\mathbf{x}]$ such that $h(\mathbf{x}) > 0$ is an interpolant for ϕ and ψ, intuitively.

Theorem 4. *Let $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$ be defined as in **Problem** 1. There must exist a polynomial $h(\mathbf{x}) \in \mathbb{Q}[\mathbf{x}]$ such that $h(\mathbf{x}) > 0$ is an interpolant for ϕ and ψ.*

Proof. We just need to prove there exists a polynomial $h(\mathbf{x}) \in \mathbb{Q}[\mathbf{x}]$ satisfying (1).

By Theorem 2, there exists a polynomial $h'(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ satisfying (1). Since $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$ are compact sets, $h'(\mathbf{x}) > 0$ on $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $h'(\mathbf{x}) < 0$ on $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$, there exist $\eta_1 > 0$ and $\eta_2 > 0$ such that

$$\forall \mathbf{x} \in P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})).h'(\mathbf{x}) - \eta_1 \geq 0,\ \forall \mathbf{x} \in P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z})).h'(\mathbf{x}) + \eta_2 \leq 0.$$

Let $\eta = \min(\frac{\eta_1}{2}, \frac{\eta_2}{2})$. Suppose $h'(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ has the form $h'(\mathbf{x}) = \sum_{\alpha \in \Omega} c_\alpha \mathbf{x}^\alpha$, where $\alpha \in \mathbb{N}^r$, $\Omega \subset \mathbb{N}^r$ is a finite set of indices, r is the dimension of $\mathbf{x}$, $\mathbf{x}^\alpha$ is the monomial $\mathbf{x}_1^{\alpha_1} \cdots \mathbf{x}_r^{\alpha_r}$, and $0 \neq c_\alpha \in \mathbb{R}$ is the coefficient of monomial $\mathbf{x}^\alpha$. Let $N = |\Omega|$ be the cardinality of Ω. Since $P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y}))$ and $P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$ are compact sets, for any $\alpha \in \Omega$, there exists $M_\alpha > 0$ such that $M_\alpha = \max\{|\mathbf{x}^\alpha| \mid \mathbf{x} \in P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})) \cup P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))\}$.

Then for any fixed polynomial $\hat{h}(\mathbf{x}) = \sum_{\alpha\in\Omega} d_\alpha \mathbf{x}^\alpha$, with $d_\alpha \in [c_\alpha - \frac{\eta}{NM_\alpha}, c_\alpha + \frac{\eta}{NM_\alpha}]$, and any $\mathbf{x} \in P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})) \cup P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z}))$, we have

$$|\hat{h}(\mathbf{x}) - h'(\mathbf{x})| = |\sum_{\alpha\in\Omega}(d_\alpha - c_\alpha)\mathbf{x}^\alpha| \le \sum_{\alpha\in\Omega}|(d_\alpha - c_\alpha)| \cdot |\mathbf{x}^\alpha| \le \sum_{\alpha\in\Omega}\frac{\eta}{NM_\alpha}\cdot M_\alpha = \eta.$$

Since $\eta = \min(\frac{\eta_1}{2}, \frac{\eta_2}{2})$, hence

$$\forall \mathbf{x} \in P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})).\hat{h}(\mathbf{x}) \ge \frac{\eta_1}{2} > 0, \quad \forall \mathbf{x} \in P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z})).\hat{h}(\mathbf{x}) \le -\frac{\eta_2}{2} < 0. \tag{7}$$

Since for any $d_\alpha \in [c_\alpha - \frac{\eta}{NM_\alpha}, c_\alpha + \frac{\eta}{NM_\alpha}]$ (7) holds, there must exist some rational number $r_\alpha \in \mathbb{Q}$ in $[c_\alpha - \frac{\eta}{NM_\alpha}, c_\alpha + \frac{\eta}{NM_\alpha}]$ satisfying (7) because of the density of rational numbers. Thus, let $h(\mathbf{x}) = \sum_{\alpha\in\Omega} r_\alpha \mathbf{x}^\alpha$. Clearly, it follows that $h(\mathbf{x}) \in \mathbb{Q}[\mathbf{x}]$ and (1) holds. □

So, the existence of $h(\mathbf{x}) \in \mathbb{Q}[\mathbf{x}]$ is guaranteed. Moreover, from the proof of Theorem 4, we know that a small perturbation of $h(\mathbf{x})$ is permitted, which is a good property for computing $h(\mathbf{x})$ in a numeric way. In the subsequent subsection, we recast the problem of finding such $h(\mathbf{x})$ as a semi-definite programming problem.

4 SOS Formulation

Similar to [7], in this section, we discuss how to reduce the problem of finding $h(\mathbf{x})$ satisfying (1) to a sum of squares programming problem.

Theorem 5. *Let $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$ be defined as in the **Problem** 1. Then there exist $m+n+2$ SOS (sum of squares) polynomials $u_i(\mathbf{x},\mathbf{y})$ $(i = 1,\ldots,m+1)$, $v_j(\mathbf{x},\mathbf{z})$ $(j = 1,\ldots,n+1)$ and a polynomial $h(\mathbf{x})$ such that*

$$h - 1 = \sum_{i=1}^{m} u_i f_i + u_{m+1}, \quad -h - 1 = \sum_{j=1}^{n} v_j g_j + v_{n+1}, \tag{8}$$

and $h(\mathbf{x}) > 0$ is an interpolant for $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$.

Proof. By Theorem 2 there exists a polynomial $\hat{h}(\mathbf{x})$ such that

$$\forall \mathbf{x} \in P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})).\hat{h}(\mathbf{x}) > 0, \quad \forall \mathbf{x} \in P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z})).\hat{h}(\mathbf{x}) < 0.$$

Set $S_1 = \{(\mathbf{x},\mathbf{y}) \mid f_1 \ge 0,\ldots,f_m \ge 0\}$ and $S_2 = \{(\mathbf{x},\mathbf{z}) \mid g_1 \ge 0,\ldots,g_n \ge 0\}$. Since $\hat{h}(\mathbf{x}) > 0$ on S_1, which is compact, there exist $\epsilon_1 > 0$ such that $\hat{h}(\mathbf{x}) - \epsilon_1 > 0$ on S_1. Similarly, there exist $\epsilon_2 > 0$ such that $-\hat{h}(\mathbf{x}) - \epsilon_2 > 0$ on S_2. Let $\epsilon = \min(\epsilon_1,\epsilon_2)$, and $h(\mathbf{x}) = \frac{\hat{h}(\mathbf{x})}{\epsilon}$, then $h(\mathbf{x}) - 1 > 0$ on S_1 and $-h(\mathbf{x}) - 1 > 0$ on S_2. Since $\mathcal{M}_{\mathbf{x},\mathbf{y}}(f_1(\mathbf{x},\mathbf{y}),\ldots,f_m(\mathbf{x},\mathbf{y}))$ is Archimedean, from Theorem 3, we have $h(\mathbf{x}) - 1 \in \mathcal{M}_{\mathbf{x},\mathbf{y}}(f_1(\mathbf{x},\mathbf{y}),\ldots,f_m(\mathbf{x},\mathbf{y}))$. Similarly, $-h(\mathbf{x}) - 1 \in \mathcal{M}_{\mathbf{x},\mathbf{z}}(g_1(\mathbf{x},\mathbf{z}),\ldots,g_n(\mathbf{x},\mathbf{z}))$. That is, there exist $m+n+2$ SOS polynomials u_i, v_j satisfying the following semi-definite constraints:

$$h(\mathbf{x}) - 1 = \sum_{i=1}^{m} u_i f_i + u_{m+1}, \quad -h(\mathbf{x}) - 1 = \sum_{j=1}^{n} v_j g_j + v_{n+1}. \quad \square$$

According to Theorem 5, the problem of finding $h(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ solving **Problem** 1 can be equivalently reformulated as the problem of searching for SOS polynomials $u_1(\mathbf{x},\mathbf{y}), \ldots, u_m(\mathbf{x},\mathbf{y}), v_1(\mathbf{x},\mathbf{z}), \ldots, v_n(\mathbf{x},\mathbf{z})$ and a polynomial $h(\mathbf{x})$ with appropriate degrees such that

$$\begin{cases} h(\mathbf{x}) - 1 - \sum_{i=1}^{m} u_i f_i \in \sum \mathbb{R}[\mathbf{x},\mathbf{y}]^2, \\ -h(\mathbf{x}) - 1 - \sum_{j=1}^{n} v_j g_j \in \sum \mathbb{R}[\mathbf{x},\mathbf{z}]^2, \\ u_i \in \sum \mathbb{R}[\mathbf{x},\mathbf{y}]^2, i = 1, \ldots, m, \\ v_j \in \sum \mathbb{R}[\mathbf{x},\mathbf{z}]^2, j = 1, \ldots, n. \end{cases} \tag{9}$$

(9) is SOS constraints over SOS multipliers $u_1(\mathbf{x},\mathbf{y}), \ldots, u_m(\mathbf{x},\mathbf{y}), v_1(\mathbf{x},\mathbf{z}), \ldots, v_n(\mathbf{x},\mathbf{z})$, polynomial $h(\mathbf{x})$, which is convex and could be solved by many existing semi-definite programming solvers such as the optimization library AiSat [7] built on CSDP [4]. Therefore, according to Theorem 5, $h(\mathbf{x}) > 0$ is an interpolant for ϕ and ψ, which is formulated in Theorem 6.

Theorem 6 (Soundness). *Suppose that $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$ are defined as in **Problem** 1, and $h(\mathbf{x})$ is a feasible solution to (9), then $h(\mathbf{x})$ solves **Problem** 1, i.e. $h(\mathbf{x}) > 0$ is an interpolant for ϕ and ψ.*

Moreover, we have the following completeness theorem stating that if the degrees of $h(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ and $u_i(\mathbf{x},\mathbf{y}) \in \sum \mathbb{R}[\mathbf{x},\mathbf{y}]^2, v_j(\mathbf{x},\mathbf{z}) \in \sum \mathbb{R}[\mathbf{x},\mathbf{z}]^2$, $i = 1, \ldots, m$, $j = 1, \ldots, n$, are large enough, $h(\mathbf{x})$ can be synthesized definitely via solving (9).

Theorem 7 (Completeness). *For **Problem** 1, there must be polynomials $u_i(\mathbf{x},\mathbf{y}) \in \mathbb{R}_N[\mathbf{x},\mathbf{y}]$ $(i = 1, \ldots, m)$, $v_j(\mathbf{x},\mathbf{z}) \in \mathbb{R}_N[\mathbf{x},\mathbf{z}]$ $(j = 1, \ldots, n)$ and $h(\mathbf{x}) \in \mathbb{R}_N[\mathbf{x}]$ satisfying (11) for some positive integer N, where $\mathbb{R}_k[\cdot]$ stands for the family of polynomials of degree no more than k.*

Proof. This is an immediate result of Theorem 5. □

Example 2. Consider two contradictory formulas ϕ and ψ defined by

$$f_1(x,y,z,a_1,b_1,c_1,d_1) \geq 0 \wedge f_2(x,y,z,a_1,b_1,c_1,d_1) \geq 0 \wedge f_3(x,y,z,a_1,b_1,c_1,d_1) \geq 0,$$
$$g_1(x,y,z,a_2,b_2,c_2,d_2) \geq 0 \wedge g_2(x,y,z,a_2,b_2,c_2,d_2) \geq 0 \wedge g_3(x,y,z,a_2,b_2,c_2,d_2) \geq 0,$$

respectively, where

$$f_1 = 4 - x^2 - y^2 - z^2 - a_1^2 - b_1^2 - c_1^2 - d_1^2, \quad f_2 = -y^4 + 2x^4 - a_1^4 - 1/100,$$
$$f_3 = z^2 - b_1^2 - c_1^2 - d_1^2 - x - 1, \quad g_1 = 4 - x^2 - y^2 - z^2 - a_2^2 - b_2^2 - c_2^2 - d_2^2,$$
$$g_2 = x^2 - y - a_2 - b_2 - d_2^2 - 3, \quad g_3 = x.$$

It is easy to observe that ϕ and ψ satisfy the conditions in **Problem** 1. Since there are local variables in ϕ and ψ and the degree of f_2 is 4, the interpolant generation methods in [7] and [10] are not applicable. We get a concrete SDP problem of the form (9) by setting the degree of the polynomial $h(x,y,z)$ in (9) to be 2. Using the MATLAB package YALMIP[23] and Mosek[28], we obtain

$$\begin{aligned}h(x,y,z) =& -416.7204 - 914.7840x + 472.6184y + 199.8985x^2 + 190.2252y^2\\ &+ 690.4208z^2 - 187.1592xy.\end{aligned}$$

Pictorially, we plot $P_{x,y,z}(\phi(x,y,z,a_1,b_1,c_1,d_1))$, $P_{x,y,z}(\psi(x,y,z,a_2,b_2,c_2,\ d_2))$ and $\{(x,y,z) \mid h(x,y,z) > 0\}$ in Fig. 2. It is evident that $h(x,y,z)$ as presented above for $d_h = 2$ is a real interpolant for $\phi(x,y,z,a,b,c,d)$ and $\psi(x,y,z,a,b,c,d)$.

5 Avoidance of the unsoundness due to numerical error in SDP

In this section, we discuss how to avoid the unsoundness of our approach caused by numerical error in SDP based on the work in [32].

A square matrix A is *positive semidefinite* if A is real symmetric and all its eigenvalues are nonnegative, denote by $A \succeq 0$.

In order to solve formula (9) to obtain $h(\mathbf{x})$, we first need to fix a degree bound of u_i, v_j and h, say $2d$, $d \in \mathbb{N}$. It is well-known that any $u(\mathbf{x}) \in \sum \mathbb{R}[\mathbf{x}]^2$ with degree $2d$ can be represented by

$$u(\mathbf{x}) \equiv E_d(\mathbf{x})^T C_u E_d(\mathbf{x}), \tag{10}$$

where $C_u \in \mathbb{R}^{\binom{r+d}{d} \times \binom{r+d}{d}}$ with $C_u \succeq 0$, $E_d(\mathbf{x})$ is a column vector with all monomials in $\mathbf{x}$, whose total degree is not greater than d, and $E_d(\mathbf{x})^T$ stands for the transposition of $E_d(\mathbf{x})$. Equaling the corresponding coefficient of each monomial whose degree is less than or equal to $2d$ at the two sides of (10), we can get a linear equation system as

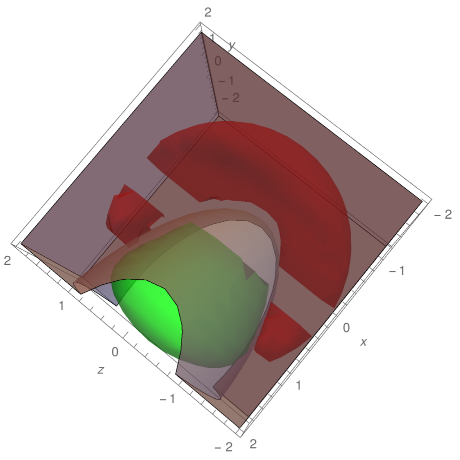

Fig. 2: Example 2. (Red region: $P_{x,y,z}(\phi(x,y,z,a_1,b_1,c_1,d_1))$; green region: $P_{x,y,z}(\psi(x,y,z,a_2,b_2,c_2,d_2))$; gray region: $\{(x,y,z) \mid h(x,y,z) > 0\}$.)

$$\mathtt{tr}(A_{u,k}C_u) = b_{u,k},\ k = 1,\ldots,K_u, \tag{11}$$

where $A_{u,k} \in \mathbb{R}^{\binom{r+d}{d} \times \binom{r+d}{d}}$ is constant matrix, $b_{u,k} \in \mathbb{R}$ is constant, $\mathtt{tr}(A)$ stands for the trace of matrix A. Thus, searching for u_i, v_j and h satisfying (9) can be reduced to the following SDP problem:

$$\begin{aligned}\mathtt{find:}\ & C_{u_1},\ldots,C_{u_m},C_{v_1},\ldots,C_{v_n},C_h,\\ \mathtt{s.t.}\ & \mathtt{tr}(A_{u_i,k}C_{u_i}) = b_{u_i,k},\ i = 1,\ldots,m, k = 1,\ldots,K_{u_i},\\ & \mathtt{tr}(A_{v_j,k}C_{v_j}) = b_{v_j,k},\ j = 1,\ldots,n, k = 1,\ldots,K_{v_j},\\ & \mathtt{tr}(A_{h,k}C_h) = b_{h,k},\ k = 1,\ldots,K_h,\\ \mathtt{diag}(& C_{u_1},\ldots,C_{u_m},C_{v_1},\ldots,C_{v_n},C_{h-1-uf},C_{-h-1-vg}) \succeq 0,\end{aligned} \tag{12}$$

where C_{h-1-uf} is the matrix corresponding to polynomial $h-1-\sum_{i=1}^{m} u_i f_i$, which is a linear combination of C_{u_1}, ..., C_{u_m} and C_h; similarly, $C_{-h-1-vg}$ is the matrix corresponding to polynomial $-h-1-\sum_{j=1}^{n} v_j g_j$, which is a linear combination of $C_{v_1},\ldots,C_{v_n}$ and C_h; and $\texttt{diag}(C_1,\ldots,C_k)$ is a block-diagonal matrix of $C_1,\ldots,C_k$.

Let D be the dimension of $C=\texttt{diag}(C_{u_1},\ldots,C_{-h-1-vg})$, i.e., $\texttt{diag}(C_{u_1},\ldots,C_{-h-1-vg})\in\mathbb{R}^{D\times D}$ and $\widehat{C}$ be the approximate solution to (12) returned by calling a numerical SDP solver, the following theorem is proved in [32].

Theorem 8 ([32], Theorem 3). *$C\succeq 0$ if there exists $\widetilde{C}\in\mathbb{F}^{D\times D}$ such that the following conditions hold: 1. $\widetilde{C}_{ij}=C_{ij}$, for any $i\neq j$; 2. $\widetilde{C}_{ii}\leq C_{ii}-\alpha$, for any i; and 3. the Cholesky algorithm implemented in floating-point arithmetic can conclude that $\widetilde{C}$ is positive semi-definite, where $\mathbb{F}$ is a floating-point format, $\alpha=\frac{(D+1)\kappa}{1-(2D+2)\kappa}\texttt{tr}(C)+4(D+1)(2(D+2)+\max_i\{C_{ii}\})\eta$, in which κ is the unit roundoff of $\mathbb{F}$ and η is the underflow unit of $\mathbb{F}$.*

Corollary 2. *Let $\widetilde{C}\in\mathbb{F}^{D\times D}$. Suppose that $\frac{(D+1)D\kappa}{1-(2D+2)\kappa}+4(D+1)\eta\leq\frac{1}{2}$, $\beta=\frac{(D+1)\kappa}{1-(2D+2)\kappa}\texttt{tr}(\widetilde{C})+4(D+1)(2(D+2)+\max_i\{\widetilde{C}_{ii}\})\eta>0$, where $\mathbb{F}$ is a floating-point format. Then $\widetilde{C}+2\beta I\succeq 0$ if the Cholesky algorithm based on floating-point arithmetic succeeds on $\widetilde{C}$, i.e., concludes that $\widetilde{C}$ is positive semi-definite.*

According to Remark 5 in [32], for IEEE 754 binary64 format with rounding to nearest, $\kappa=2^{-53}(\simeq 10^{-16})$ and $\eta=2^{-1075}(\simeq 10^{-323})$. In this case, the order of magnitude of β is 10^{-10} and $\frac{(D+1)D\kappa}{1-(2D+2)\kappa}+4(D+1)\eta$ is 10^{-13}, much less than $\frac{1}{2}$. Obviously, β becomes smaller when the length of binary format becomes longer. W.l.o.g., we suppose that the Cholesky algorithm succeed in computing $\widehat{C}$ the solution of (12), which is reasonable as if an SDP solver returns a solution $\widehat{C}$, then $\widehat{C}$ should be considered to be positive semi-definite in the sense of numeric computation.

So, by Corollary 2, we have $\widehat{C}+2\beta I\succeq 0$ holds, where I is the identity matrix with the corresponding dimension. Then we have

$$\texttt{diag}(\widehat{C}_{u_1},\ldots,\widehat{C}_{u_m},\widehat{C}_{v_1},\ldots,\widehat{C}_{v_n},\widehat{C}_{h-1-uf},\widehat{C}_{-h-1-vg})+2\beta I\succeq 0.$$

Let $\epsilon=\max_{p\in P,1\leq i\leq K_p}|\texttt{tr}(A_{p,i}\widehat{C}_p)-b_{p,i}|$, where $P=\{u_1,\ldots,u_m,v_1,\ldots,v_n,h\}$, which can be regarded as the tolerance of the SDP solver. Since $|\texttt{tr}(A_{p,i}C_p)-b_{p,i}|$ is the error term for each monomial of p, i.e., ϵ can be considered as the error bound on the coefficients of polynomials u_i, v_j and h, for any polynomial $\hat{u}_i$ ($\hat{v}_j$ and $\hat{h}$), computed from (11) by replacing C_u with the corresponding $\widehat{C_u}$, there exists a corresponding remainder term R_{u_i} (resp. R_{v_j} and R_h) with degree not greater than $2d$, whose coefficients are bounded by ϵ. Hence, we have

$$\begin{aligned}
&\widehat{u_i}+R_{u_i}+2\beta E_d(\mathbf{x},\mathbf{y})^T E_d(\mathbf{x},\mathbf{y})\in\sum\mathbb{R}[\mathbf{x},\mathbf{y}]^2, i=1,\ldots,m,\\
&\widehat{v_j}+R_{v_j}+2\beta E_d(\mathbf{x},\mathbf{z})^T E_d(\mathbf{x},\mathbf{z})\in\sum\mathbb{R}[\mathbf{x},\mathbf{z}]^2, j=1,\ldots,n,\\
&\widehat{h}+R_h-1-\sum_{i=1}^{m}(\widehat{u_i}+R'_{u_i})f_i+2\beta E_d(\mathbf{x},\mathbf{y})^T E_d(\mathbf{x},\mathbf{y})\in\sum\mathbb{R}[\mathbf{x},\mathbf{y}]^2,
\end{aligned}\tag{13}$$

$$-\widehat{h}+R'_h-1-\sum_{j=1}^{m}(\widehat{v_j}+R'_{v_j})g_j+2\beta E_d(\mathbf{x},\mathbf{z})^T E_d(\mathbf{x},\mathbf{z})\in\sum\mathbb{R}[\mathbf{x},\mathbf{z}]^2.$$

Now, in order to avoid unsoundness of our approach caused by the numerical issue due to SDP, we have to prove

$$f_1\geq 0\wedge\cdots\wedge f_m\geq 0\Rightarrow\widehat{h}>0, \tag{14}$$

$$g_1\geq 0\wedge\cdots\wedge g_n\geq 0\Rightarrow\widehat{h}<0. \tag{15}$$

Regarding (14), let $R_{2d,\mathbf{x}}$ be a polynomial in $\mathbb{R}[|\mathbf{x}|]$, whose total degree is $2d$, and all coefficients are 1, e.g., $R_{2,x,y}=1+|x|+|y|+|x^2|+|xy|+|y^2|$. Since $S=\{(\mathbf{x},\mathbf{y})\mid f_1\geq 0\wedge\cdots\wedge f_m\geq 0\}$ is a compact set, then for any polynomial $p\in\mathbb{R}[\mathbf{x},\mathbf{y}]$, $|p|$ is bounded on S. Let M_1 be an upper bound of $R_{2d,\mathbf{x},\mathbf{y}}$ on S, M_2 an upper bound of $E_d(\mathbf{x},\mathbf{y})^T E_d(\mathbf{x},\mathbf{y})$, and M_{f_i} an upper bound of f_i on S. Then, $|R_{u_i}|$, $|R'_{u_i}|$ and $|R_h|$ are bounded by ϵM_1. Let $E_{\mathbf{xy}}=E_d(\mathbf{x},\mathbf{y})^T E_d(\mathbf{x},\mathbf{y})$. So for any $(\mathbf{x}_0,\mathbf{y}_0)\in S$, considering the polynomials below at $(\mathbf{x}_0,\mathbf{y}_0)\in S$, by the first and third line in (13),

$$\begin{aligned}
\widehat{h}\geq & 1-R_h+\sum_{i=1}^{m}(\widehat{u_i}+R'_{u_i})f_i-2\beta E_{\mathbf{xy}}\\
\geq & 1-\epsilon M_1+\sum_{i=1}^{m}(\widehat{u_i}+R_{u_i}+2\beta E_{\mathbf{xy}}+R'_{u_i}-R_{u_i}-2\beta E_{xy})f_i-2\beta M_2\\
= & 1-\epsilon M_1-2\beta M_2+\sum_{i=1}^{m}(\widehat{u_i}+R_{u_i}+2\beta E_{\mathbf{xy}})f_i+\sum_{i=1}^{m}(R'_{u_i}-R_{u_i}-2\beta E_{\mathbf{xy}})f_i\\
\geq & 1-\epsilon M_1-2\beta M_2+0-\sum_{i=1}^{m}(\epsilon M_1+\epsilon M_1+2\beta M_2)M_{f_i}\\
= & 1-(2\sum_{i=1}^{m}M_{f_i}+1)M_1\epsilon-2(\sum_{i=1}^{m}M_{f_i}+1)M_2\beta.
\end{aligned}$$

Whence,

$$f_1\geq 0\wedge\cdots\wedge f_m\geq 0\Rightarrow\widehat{h}\geq 1-(2\sum_{i=1}^{m}M_{f_i}+1)M_1\epsilon-2(\sum_{i=1}^{m}M_{f_i}+1)M_2\beta.$$

Let $S'=\{(\mathbf{x},\mathbf{z})\mid g_1\geq 0\wedge\cdots\wedge g_n\geq 0\}$, M_3 be an upper bound of $R_{2d,\mathbf{x},\mathbf{z}}$ on S', M_4 an upper bound of $E_d(\mathbf{x},\mathbf{z})^T E_d(\mathbf{x},\mathbf{z})$ on S', and M_{g_j} an upper bound of g_j on S'. Similarly, it follows

$$g_1\geq 0\wedge\cdots\wedge g_n\geq 0\Rightarrow\quad -\widehat{h}\geq 1-(2\sum_{j=1}^{n}M_{g_j}+1)M_3\epsilon-2(\sum_{j=1}^{n}M_{g_j}+1)M_4\beta.$$

So, the following proposition is immediately.

Proposition 2. *There exist two positive constants γ_1 and γ_2 such that*

$$f_1\geq 0\wedge\cdots\wedge f_m\geq 0\Rightarrow\widehat{h}\geq 1-\gamma_1\epsilon-\gamma_2\beta, \tag{16}$$

$$g_1\geq 0\wedge\cdots\wedge g_n\geq 0\Rightarrow-\widehat{h}\geq 1-\gamma_1\epsilon-\gamma_2\beta. \tag{17}$$

Since ϵ and β heavily rely on the numerical tolerance and the floating point representation, it is easy to see that ϵ and β become small enough with $\gamma_1\epsilon < \frac{1}{2}$ and $\gamma_2\beta < \frac{1}{2}$, if the numerical tolerance is small enough and the length of the floating point representation is long enough. This implies

$$f_1 \geq 0 \wedge \cdots \wedge f_m \geq 0 \Rightarrow \widehat{h} > 0, \quad g_1 \geq 0 \wedge \cdots \wedge g_n \geq 0 \Rightarrow -\widehat{h} > 0.$$

If so, any numerical result $\widehat{h} > 0$ returned by calling an SDP solver to (12) is guaranteed to be a real interpolant for ϕ and ψ, i.e., a correct solution to **Problem** 1.

Example 3. Consider the numerical result for Example 2 in Section 4. Let M_{f_1}, M_{f_2}, M_{f_3}, M_{g_1}, M_{g_2}, M_{g_3}, M_1, M_2, M_3, M_4 are defined as above. It is easy to see that

$$f_1 \geq 0 \Rightarrow |x| \leq 2 \wedge |y| \leq 2 \wedge |z| \leq 2 \wedge |a_1| \leq 2 \wedge |b_1| \leq 2 \wedge |c_1| \leq 2 \wedge |d_1| \leq 2.$$

Then, by simple calculations, we obtain $M_{f_1} = 4, M_{f_2} = 32, M_{f_3} = 3, M_1 = 83, M_2 = 29$. Thus,

$$(2\sum_{i=1}^{m} M_{f_i} + 1)M_1 = 6557, \quad 2(\sum_{i=1}^{m} M_{f_i} + 1)M_2 = 2320.$$

Also, since

$$g_1 \geq 0 \Rightarrow |x| \leq 2 \wedge |y| \leq 2 \wedge |z| \leq 2 \wedge |a_2| \leq 2 \wedge |b_2| \leq 2 \wedge |c_2| \leq 2 \wedge |d_2| \leq 2,$$

we obtain $M_{g_1} = 4, M_{g_2} = 7, M_{g_3} = 2, M_3 = 83, M_4 = 29$. Thus,

$$(2\sum_{i=1}^{m} M_{g_i} + 1)M_3 = 2241, \quad 2(\sum_{i=1}^{m} M_{g_i} + 1)M_4 = 812.$$

Consequently, we have $\gamma_1 = 6557$ and $\gamma_2 = 2320$ in Proposition 2.

Due to the fact that the default error tolerance is 10^{-8} in the SDP solver Mosek and h is rounding to 4 decimal places, we have $\epsilon = \frac{10^{-4}}{2}$. In addition, as the absolute value of each element in $\widehat{C}$ is less than 10^3, and the dimension of D is less than 10^3, we obtain

$$\beta = \frac{(D+1)\kappa}{1-(2D+2)\kappa}\mathtt{tr}(\widetilde{C}) + 4(D+1)(2(D+2) + \max_i(\widetilde{C}_{ii}))\eta \leq 10^{-6}.$$

Consequently, $\gamma_1\epsilon \leq 6557 \cdot \frac{10^{-4}}{2} < \frac{1}{2}$, $\gamma_2\beta \leq 2320 \cdot 10^{-6} < \frac{1}{2}$, which imply that $h(x,y,z) > 0$ presented in Example 2 is indeed a real interpolant.

Remark 1. Besides, the result could be verified by the following symbolic computation procedure instead: computing $P_{\mathbf{x}}(\phi)$ and $P_{\mathbf{x}}(\psi)$ first by some symbolic tools, such as Redlog [8] which is a package that extends the computer algebra system REDUCE to a computer logic system; then verifying $\mathbf{x} \in P_{\mathbf{x}}(\phi) \Rightarrow h(\mathbf{x}) > 0$ and $\mathbf{x} \in P_{\mathbf{x}}(\psi) \Rightarrow h(\mathbf{x}) < 0$. For this example, $P_{x,y,z}(\phi)$ and $P_{x,y,z}(\psi)$ obtained by Redlog are too complicated and therefore not presented here. The symbolic computation can verify that $h(x,y,z)$ in this example is exactly an interpolant, which confirms our conclusion. Alternatively, we can also solve the SDP in (9) using a SDP solver with infinite precision [15], and obtain an exact result. But this only works for problems with small size because a SDP solver with infinite precision is essentially based on symbolic computation as commented in [15].

6 Generalizing to general polynomial formulas

Problem 2. Let $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$ be two polynomial formulas defined as follows,

$$\phi(\mathbf{x},\mathbf{y}) : \bigvee_{i=1}^{m} \phi_i,\ \phi_i = \bigwedge_{k=1}^{K_i} f_{i,k}(\mathbf{x},\mathbf{y}) \geq 0; \quad \psi(\mathbf{x},\mathbf{z}) : \bigvee_{j=1}^{n} \psi_j,\ \psi_j = \bigwedge_{s=1}^{S_j} g_{j,s}(\mathbf{x},\mathbf{z}) \geq 0,$$

where all $f_{i,k}$ and $g_{j,s}$ are polynomials. Suppose $\phi \wedge \psi \models \perp$, and for $i = 1,\ldots,m$, $j = 1,\ldots,n$, $\{(\mathbf{x},\mathbf{y}) \mid \phi_i(\mathbf{x},\mathbf{y})\}$ and $\{(\mathbf{x},\mathbf{z}) \mid \psi_j(\mathbf{x},\mathbf{z})\}$ are all semi-algebraic sets of the Archimedean form. Find a polynomial $h(\mathbf{x})$ such that $h(\mathbf{x}) > 0$ is an interpolant for ϕ and ψ.

Theorem 9. *For* **Problem** *2, there exists a polynomial $h(\mathbf{x})$ satisfying*

$$\forall \mathbf{x} \in P_{\mathbf{x}}(\phi(\mathbf{x},\mathbf{y})).h(\mathbf{x}) > 0, \quad \forall \mathbf{x} \in P_{\mathbf{x}}(\psi(\mathbf{x},\mathbf{z})).h(\mathbf{x}) < 0.$$

Proof. We just need to prove that Lemma 1 holds for **Problem** 2 as well. Since $\{(\mathbf{x},\mathbf{y}) \mid \phi_i(\mathbf{x},\mathbf{y})\}$ and $\{(\mathbf{x},\mathbf{z}) \mid \psi_j(\mathbf{x},\mathbf{z})\}$ are all semi-algebraic sets of the Archimedean form, then $\{(\mathbf{x},\mathbf{y}) \mid \phi(\mathbf{x},\mathbf{y})\}$ and $\{(\mathbf{x},\mathbf{z}) \mid \psi(\mathbf{x},\mathbf{z})\}$ both are compact. See $\{(\mathbf{x},\mathbf{y}) \mid \phi(\mathbf{x},\mathbf{y})\}$ or $\{(\mathbf{x},\mathbf{z}) \mid \psi(\mathbf{x},\mathbf{z})\}$ as S in the proof of Lemma 1, then Lemma 1 holds for **Problem** 2. Thus, the rest of proof is same as that for Theorem 2. $\square$

Corollary 3. *Let $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$ be defined as in* **Problem** *2. There must exist a polynomial $h(\mathbf{x})$ such that $h(\mathbf{x}) > 0$ is an interpolant for ϕ and ψ.*

Theorem 10. *Let $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$ be defined as in* **Problem** *2. Then there exists a polynomial $h(\mathbf{x})$ and $\sum_{i=1}^{m}(K_i+1)+\sum_{j=1}^{n}(S_j+1)$ sum of squares polynomials $u_{i,k}(\mathbf{x},\mathbf{y})$ ($i = 1,\ldots,m$, $k = 1,\ldots,K_i+1$), $v_{j,s}(\mathbf{x},\mathbf{z})$ ($j = 1,\ldots,n$, $s = 1,\ldots,S_j$) satisfying the following semi-definite constraints such that $h(\mathbf{x}) > 0$ is an interpolant for $\phi(\mathbf{x},\mathbf{y})$ and $\psi(\mathbf{x},\mathbf{z})$:*

$$h - 1 = \sum_{k=1}^{K_i} u_{i,k} f_{i,k} + u_{i,K_i+1}, \quad i = 1,\ldots,m; \tag{18}$$

$$-h - 1 = \sum_{s=1}^{S_j} v_{j,s} g_{j,s} + v_{j,S_j+1}, \quad j = 1,\ldots,n. \tag{19}$$

Proof. By the property of Archimedean, the proof is same as that for Theorem 5. $\square$

Similarly, **Problem** 2 can be equivalently reformulated as the problem of searching for sum of squares polynomials satisfying

$$\begin{cases} h(\mathbf{x}) - 1 - \sum_{k=1}^{K_i} u_{i,k} f_{i,k} \in \sum \mathbb{R}[\mathbf{x},\mathbf{y}]^2, i = 1,\ldots,m; \\ -h(\mathbf{x}) - 1 - \sum_{s=1}^{S_j} v_{j,s} g_{j,s} \in \sum \mathbb{R}[\mathbf{x},\mathbf{z}]^2, j = 1,\ldots,n; \\ u_{i,k} \in \sum \mathbb{R}[\mathbf{x},\mathbf{y}]^2, i = 1,\ldots,m, k = 1,\ldots,K_i; \\ v_{j,s} \in \sum \mathbb{R}[\mathbf{x},\mathbf{z}]^2, j = 1,\ldots,n, s = 1,\ldots,S_j. \end{cases} \tag{20}$$

Example 4. Consider

$$\begin{aligned}\phi(x,y,a_1,a_2,b_1,b_2) &: (f_1 \geq 0 \wedge f_2 \geq 0) \vee (f_3 \geq 0 \wedge f_4 \geq 0),\\ \psi(x,y,c_1,c_2,d_1,d_2) &: (g_1 \geq 0 \wedge g_2 \geq 0) \vee (g_3 \geq 0 \wedge g_4 \geq 0),\end{aligned}$$

where

$$\begin{aligned}f_1 &= 16-(x+y-4)^2-16(x-y)^2-a_1^2, & f_2 &= x+y-a_2^2-(2-a_2)^2,\\ f_3 &= 16-(x+y+4)^2-16(x-y)^2-b_1^2, & f_4 &= -x-y-b_2^2-(2-b_2)^2,\\ g_1 &= 16-16(x+y)^2-(x-y+4)^2-c_1^2, & g_2 &= y-x-c_2^2-(1-c_2)^2,\\ g_3 &= 16-16(x+y)^2-(x-y-4)^2-d_1^2, & g_4 &= x-y-d_2^2-(1-d_2)^2.\end{aligned}$$

We get a concrete SDP problem of the form (20) by setting the degree of $h(x,y)$ in (20) to be 2. Using the MATLAB package YALMIP and Mosek, we obtain

$$h(x,y) = -2.3238 + 0.6957x^2 + 0.6957y^2 + 7.6524xy.$$

The result is plotted in Fig. 3, and can be verified either by numerical error analysis as in Example 2 or by a symbolic procedure like REDUCE as described in Remark 1.

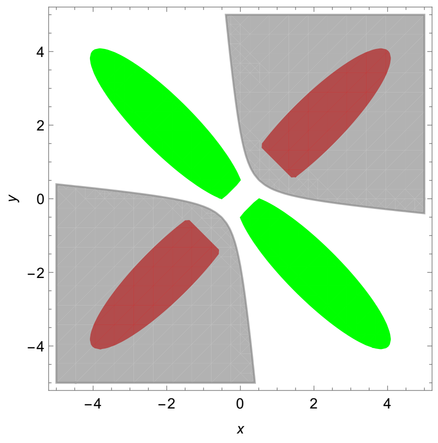

Fig. 3: Example 4. (Red region: $P_{x,y}(\phi(x,y,a_1,a_2,b_1,b_2))$; green region: $P_{x,y}(\psi(x,y,c_1,c_2,d_1,d_2))$; gray region: $\{(x,y) \mid h(x,y) > 0\}$.)

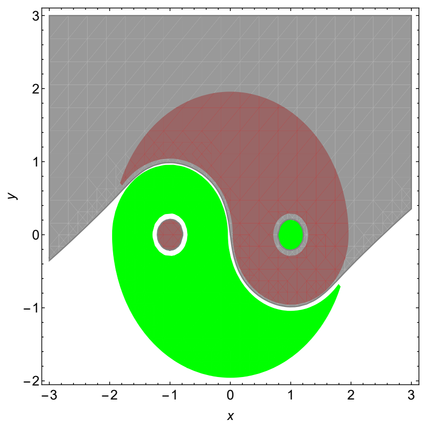

Fig. 4: Example 5. (Red region: $P_{x,y}(\phi(x,y))$; green region: $P_{x,y}(\psi(x,y))$; gray region: $\{(x,y) \mid h(x,y) > 0\}$.)

Example 5 (Ultimate). Consider the following example taken from [5], which is a challenging benchmark to existing approaches for nonlinear interpolant generation.

$$\phi = (f_1 \geq 0 \wedge f_2 \geq 0 \vee f_3 \geq 0) \wedge f_4 \geq 0 \wedge f_5 \geq 0 \vee f_6 \geq 0,$$

$$\psi = (g_1 \geq 0 \wedge g_2 \geq 0 \vee g_3 \geq 0) \wedge g_4 \geq 0 \wedge g_5 \geq 0 \vee g_6 \geq 0,$$

where

$$\begin{array}{ll} f_1 = 3.8025 - x^2 - y^2, & f_2 = y, \\ f_3 = 0.9025 - (x-1)^2 - y^2, & f_4 = (x-1)^2 + y^2 - 0.09, \\ f_5 = (x+1)^2 + y^2 - 1.1025, & f_6 = 0.04 - (x+1)^2 - y^2, \\ g_1 = 3.8025 - x^2 - y^2, & g_2 = -y, \\ g_3 = 0.9025 - (x+1)^2 - y^2, & g_4 = (x+1)^2 + y^2 - 0.09, \\ g_5 = (x-1)^2 + y^2 - 1.1025, & g_6 = 0.04 - (x-1)^2 - y^2. \end{array}$$

We first convert ϕ and ψ to the disjunction normal form as:

$$\phi = (f_1 \geq 0 \wedge f_2 \geq 0 \wedge f_4 \geq 0 \wedge f_5 \geq 0) \vee (f_3 \geq 0 \wedge f_4 \geq 0 \wedge f_5 \geq 0) \vee (f_6 \geq 0),$$
$$\psi = (g_1 \geq 0 \wedge g_2 \geq 0 \wedge g_4 \geq 0 \wedge g_5 \geq 0) \vee (g_3 \geq 0 \wedge g_4 \geq 0 \wedge g_5 \geq 0) \vee (g_6 \geq 0).$$

We get a concrete SDP problem of the form (20) by setting the degree of $h(x,y)$ in (20) to be 7. Using the MATLAB package YALMIP and Mosek, keeping the decimal to four, we obtain

$$\begin{aligned} h(x,y) = {} & 1297.5980x + 191.3260y - 3172.9653x^3 + 196.5763x^2y + 2168.1739xy^2 + \\ & 1045.7373y^3 + 1885.8986x^5 - 1009.6275x^4y + 3205.3793x^3y^2 - 1403.5431x^2y^3 \\ & + 1842.0669xy^4 + 1075.2003y^5 - 222.0698x^7 + 547.9542x^6y - 704.7474x^5y^2 \\ & + 1724.7008x^4y^3 - 728.2229x^3y^4 + 1775.7548x^2y^5 - 413.3771xy^6 + 1210.2617y^7. \end{aligned}$$

The result is plotted in Fig. 4, and can be verified either by numerical error analysis as in Example 2 or by a symbolic procedure like REDUCE as described in Remark 1.

7 Application to Invariant Generation

In this section, as an application, we sketch how to apply our approach to invariant generation in program verification, the details can be found in [11].

In [22], Lin *et al.* proposed a framework for invariant generation using *weakest precondition*, *strongest postcondition* and *interpolation*, which consists of two procedures, i.e., synthesizing invariants by forward interpolation based on *strongest postcondition* and *interpolant generation*, and by backward interpolation based on *weakest precondition* and *interpolant generation*. In [22], only linear invariants can be synthesized as no powerful approaches are available to synthesize nonlinear interpolants. Obviously, our results can strengthen their framework by allowing to generate nonlinear invariants. For example, we can revise the procedure Squeezing Invariant - Forward in their framework and obtain Algorithm 1. The major revisions include:

- firstly, we exploit our method to synthesize interpolants see line 4 in Algorithm 1;
- secondly, we add a conditional statement for A_{i+1} at line 7-10 in Algorithm 1 in order to make A_{i+1} to be Archimedean.

The procedure Squeezing Invariant - backward can be revised similarly.

Example 6. Consider a loop program given in Algorithm 2 for controlling the acceleration of a car adapted from [21]. Suppose we know that *vc* is in $[0, 40]$ at the beginning

Algorithm 1 Revised Squeezing Invariant - Forward

```
Input: An annotated loop: $\{P\}$ while $\rho$ do $C$ $\{Q\}$, where $P$ and $Q$ are Archimedean
Output: (yes/no, $\mathcal{I}$), where $\mathcal{I}$ is a loop invariant
 1: $A_0 \leftarrow P$; $B_0 \leftarrow (\neg\rho \wedge \neg Q)$; $i \leftarrow 0$; $j \leftarrow 0$;
 2: while $\top$ do
 3:   if $(\bigvee_{k=0}^{i} A_i) \wedge B_j$ is not satisfiable, $(\bigvee_{k=0}^{i} A_i)$ and $B_j$ are Archimedean then
 4:     call our method to synthesize an interpolant for $(\bigvee_{k=0}^{i} A_i)$ and $B_j$, say $\mathcal{I}_i$;
        $\{$Use our method to generate interpolant$\}$
 5:     if $\{\mathcal{I}_i \wedge \rho\}\, C\, \{\mathcal{I}_i\}$ then
 6:       return (yes, $\mathcal{I}_i$);
 7:     else if $\mathcal{I}_i$ is Archimedean then
 8:       $A_{i+1} \leftarrow \mathsf{sp}(\mathcal{I}_i \wedge \rho, C)$;
 9:     else
10:       $A_{i+1} \leftarrow \mathsf{sp}(A_i \wedge \rho, C)$;
11:     end if
        $\{\mathsf{sp}$: a predicate transformer to compute the strongest postcondition of $C$ w.r.t. $\mathcal{I}_i \wedge \rho\}$
12:     $i \leftarrow i+1$;  $B_{j+1} \leftarrow B_0 \vee (\rho \wedge \mathsf{wp}(C, B_j))$;
        $\{\mathsf{wp}$: a predicate transformer to compute the weakest precondition of $C$ w.r.t. $B_j\}$
13:     $j \leftarrow j+1$;
14:   else if $A_i$ is concrete then
15:     return (no, $\perp$);
16:   else
17:     while $A_i$ is not concrete do
18:       $i \leftarrow i-1$;
19:     end while
20:     $A_{i+1} \leftarrow \mathsf{sp}(A_i \wedge \rho, C)$;  $i \leftarrow i+1$;
21:   end if
22: end while
```

of the loop, we would like to prove that $vc < 49.61$ holds after the loop. Since the loop guard is unknown, it means that the loop may terminate after any number of iterations.

We apply Algorithm 1 to the computation of an invariant to ensure that $vc < 49.61$ holds. Since vc is the velocity of car, $0 \leq vc < 49.61$ is required to hold in order to maintain safety. Via Algorithm 1, we have $A_0 = \{vc \mid vc(40 - vc) \geq 0\}$ and $B = \{vc \mid vc < 0\} \cup \{vc \mid vc \geq 49.61\}$. Here, we replace B with $B' = [-2, -1] \cup [49.61, 55])$, i.e., $B' = \{vc \mid (vc+2)(-1-vc) \geq 0 \vee (vc - 49.61)(55 - vc) \geq 0\}$, in order to make it with Archimedean form.

Firstly, it is evident that $A_0 : vc(40 - vc) \geq 0$ implies $A_0 \wedge B' \models \perp$. By applying our approach, we obtain an interpolant

$$\mathcal{I}_0 : 1.4378 + 3.3947 * vc - 0.083 * vc^2 > 0$$

for A_0 and B'. It can be verified that $\{\mathcal{I}_0\}\, C\, \{\mathcal{I}_0\}$ (line 5) does not hold, where C stands for the loop body.

Secondly, by setting $A_1 = sp(\mathcal{I}_0, C)$ (line 8) and re-calling our approach, we obtain an interpolant

$$\mathcal{I}_1 : 2.0673 + 3.0744 * vc - 0.0734 * vc^2 > 0$$

for $A_0 \cup A_1$ and B'. Likewise, it can be verified that $\{\mathcal{I}_1\}\, C\, \{\mathcal{I}_1\}$ (line 5) does not hold.

Algorithm 2 Control code for accelerating a car

```
1: /* Pre: $vc \in [0, 40]$ */
2: while unknown do
3:     $fa \leftarrow 0.5418 * vc * vc$;
4:     $fr \leftarrow 1000 - fa$;
5:     $ac \leftarrow 0.0005 * fr$;
6:     $vc \leftarrow vc + ac$;
7: end while
8: /* Post: $vc < 49.61$ */
```

Thirdly, repeating the above procedure again, we obtain an interpolant

$$\mathcal{I}_2 : 2.2505 + 2.7267 * vc - 0.063 * vc^2 > 0,$$

and it can be verified that $\{\mathcal{I}_2\}\, C\, \{\mathcal{I}_2\}$ holds, implying that $\mathcal{I}_2$ is an invariant. Moreover, it is trivial to verify that $\mathcal{I}_2 \Rightarrow vc < 49.61$.

Consequently, we have the conclusion that $\mathcal{I}_2$ is an inductive invariant which witnesses the correctness of the loop.

8 Conclusion

In this paper we propose a sound and complete method to synthesize Craig interpolants for mutually contradictory polynomial formulas $\phi(\mathbf{x}, \mathbf{y})$ and $\psi(\mathbf{x}, \mathbf{z})$, with the form $f_1 \geq 0 \wedge \cdots \wedge f_n \geq 0$, where f_i's are polynomials in $\mathbf{x}, \mathbf{y}$ or $\mathbf{x}, \mathbf{z}$ and the quadratic module generated by f_i's is Archimedean. The interpolant is generated by solving a semi-definite programming problem, which is a generalization of the method in [7] dealing with mutually contradictory formulas with the same set of variables and the method in [10] dealing with mutually contradictory formulas with concave quadratic polynomial inequalities. As an application, we apply our approach to invariant generation in program verification.

As a future work, we would like to consider interpolant synthesizing for formulas with strict polynomial inequalities. Also, it deserves to consider how to synthesize interpolants for the combination of non-linear formulas and other theories based on our approach and other existing ones, as well as further applications to the verification of programs and hybrid systems.

Acknowledgments

We thank Dr. Sicun Gao, Dr. Damien Zufferey and Dr. Mingshuai Chen for their help on using dReal. We are indebted to anonymous reviewers for their detailed and constructive criticisms and comments on the preliminary version, which help to improve the presentation of this paper very much. This work is financially supported by NSFC under grants 61902284 (the first author), 61732001 (the second and fourth authors), 61532019 (the second author), 61836005 (the third author), 61625206 (the fourth author) and 61802318 (the fifth author), and the third author is also in part supported by Hundred Talents Program under grant No. Y8YC235015.

References

1. F. Benhamou and L. Granvilliers. Continuous and interval constraints. In *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*, pages 571–603. 2006.
2. E. Bierstone and P. D. Milman. Semianalytic and subanalytic sets. *Publications Mathematiques de l'IHÉS*, 67:5–42, 1988.
3. J. Bochnak, M. Coste, and M. Roy. *Real Algebraic Geometry*. Springer,, 1998.
4. B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods & Software*, 11(1-4):613–623, 1999. http://projects.coin-or.org/csdp/.
5. M. Chen, J. Wang, J. An, B. Zhan, D. Kapur, and N. Zhan. NIL: learning nonlinear interpolants. In *CADE 2019*, volume 11716 of *Lecture Notes in Computer Science*, pages 178–196, 2019.
6. A. Cimatti, A. Griggio, and R. Sebastiani. Efficient interpolation generation in satisfiability modulo theories. In *TACAS 2008*, volume 4963 of *Lecture Notes in Computer Science*, pages 397–412, 2008.
7. L. Dai, B. Xia, and N. Zhan. Generating non-linear interpolants by semidefinite programming. In *CAV 2013*, volume 8044 of *Lecture Notes in Computer Science*, pages 364–380, 2013.
8. A. Dolzmann and T. Sturm. REDLOG: computer algebra meets computer logic. *ACM SIGSAM Bulletin*, 31(2):2–9, 1997.
9. V. D'Silva, M. Purandare, G. Weissenbacher, and D. Kroening. Interpolant strength. In *VMCAI 2010*, volume 5944 of *Lecture Notes in Computer Science*, pages 129–145, 1997.
10. T. Gan, L. Dai, Xia B, N. Zhan, D. Kapur, and M. Chen. Interpolation synthesis for quadratic polynomial inequalities and combination with *EUF*. In *IJCAR 2016*, volume 9706 of *Lecture Notes in Computer Science*, pages 195–212, 2016.
11. T. Gan, B. Xia, B. Xue, and N. Zhan. Nonlinear craig interpolant generation. *CoRR*, abs/1903.01297, 2019.
12. S. Gao, S. Kong, and E. Clarke. Proof generation from delta-decisions. In *SYNASC 2014*, pages 156–163, 2014.
13. S. Gao and D. Zufferey. Interpolants in nonlinear theories over the reals. In *TACAS 2016*, volume 9636 of *Lecture Notes in Computer Science*, pages 625–641, 2016.
14. S. Graf and H. Saïdi. Construction of abstract state graphs with PVS. In *CAV 1997*, volume 1254 of *Lecture Notes in Computer Science*, pages 72–83, 1997.
15. D. Henrion, S. Naldi, and M. Safey El Din. Exact algorithms for semidefinite programs with degenerate feasible set. In *ISSAC 2018*, pages 191–198, 2018.
16. T. Henzinger, R. Jhala, R. Majumdar, and K. McMillan. Abstractions from proofs. In *POPL 2004*, pages 232–244, 2004.
17. Y. Jung, W. Lee, B. Wang, and K. Yi. Predicate generation for learning-based quantifier-free loop invariant inference. In *TACAS 2011*, volume 6605 of *Lecture Notes in Computer Science*, pages 205–219, 2011.
18. D. Kapur, R. Majumdar, and C. Zarba. Interpolation for data structures. In *FSE 2006*, pages 105–116, 2006.
19. L. Kovács and A. Voronkov. Interpolation and symbol eleimination. In *CADE 2009*, volume 5663 of *Lecture Notes in Computer Science*, pages 199–213, 2009.
20. J. Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *J. of Symbolic Logic*, 62(2):457–486, 1997.
21. S. Kupferschmid and B. Becker. Craig interpolation in the presence of non-linear constraints. In *FORMATS 2011*, volume 6919 of *Lecture Notes in Computer Science*, 2011.

22. S. Lin, J. Sun, H. Xiao, D. Sanán, and H. Hansen. Fib: Squeezing loop invariants by interpolation between forward/backward predicate transformers. In *ASE 2017*, pages 793–803, 2017.
23. J. Lofberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *CACSD 2004*, pages 284–289. IEEE, 2004.
24. M. Marshall. *Positive Polynomials and Sums of Squares*. American Mathematical Society,, 2008.
25. K. McMillan. Interpolation and sat-based model checking. In *CAV 2003*, volume 3920 of *Lecture Notes in Computer Science*, pages 1–13, 2003.
26. K. McMillan. An interpolating theorem prover. *Theor. Comput. Sci.*, 345(1):101–121, 2005.
27. K. McMillan. Quantified invariant generation using interpolation saturation prover. In *TACAS 2008*, volume 4963 of *Lecture Notes in Computer Science*, pages 413–427, 2008.
28. A. Mosek. The MOSEK optimization toolbox for MATLAB manual. *Version 7.1 (Revision 28)*, page 17, 2015.
29. T. Mostowski. Some properties of the ring of nash functions. *Annali della Scuola Normale Superiore di Pisa*, 3(2):245–266, 1976.
30. P. Pudlăk. Lower bounds for resolution and cutting plane proofs and monotone computations. *J. of Symbolic Logic*, 62(3):981–998, 1997.
31. M. Putinar. Positive polynomials on compact semi-algebraic sets. *Indiana University Mathematics Journal*, 42(3):969–984, 1993.
32. P. Roux, Y. L. Voronin, and S. Sankaranarayanan. Validating numerical semidefinite programming solvers for polynomial invariants. *Formal Methods in System Design*, (4):1–27, 2016.
33. A. Rybalchenko and V. Sofronie-Stokkermans. Constraint solving for interpolation. *J. Symb. Comput.*, 45(11):1212–1233, 2010.
34. V. Sofronie-Stokkermans. Interpolation in local theory extensions. *Logical Methods in Computer Science*, 4(4), 2008.
35. A. Srikanth, B. Sahin, and W. Harris. Complexity verification using guided theorem enumeration. In *POPL 2017*, pages 639–652, 2017.
36. G. Stengle. A nullstellensatz and a positivstellensatz in semialgebraic geometry. *Ann. Math.*, 207:87–97, 1974.
37. G. Yorsh and M. Musuvathi. A combination method for generating interpolants. In *CADE 2005*, volume 3632 of *Lecture Notes in Artificial Intelligence*, pages 353–368, 2005.
38. N. Zhan, S. Wang, and H. Zhao. *Formal Verification Simulink/Stateflow Diagrams: A Deductive Way*. Springer, 2017.
39. H. Zhao, N. Zhan, D. Kapur, and K. Larsen. A "hybrid" approach for synthesizing optimal controllers of hybrid systems: A case study of the oil pump industrial example. In *FM 2012*, volume 7436 of *Lecture Notes in Computer Science*, pages 471–485. 2012.