Abstract Partial Cylindrical Algebraic Decomposition I: The Lifting Phase*

Grant Olney Passmore[1,2] and Paul B. Jackson[2]
grant.passmore@cl.cam.ac.uk, pbj@inf.ed.ac.uk

[1] Clare Hall, University of Cambridge
[2] LFCS, University of Edinburgh

Abstract. Though decidable, the theory of real closed fields (**RCF**) is fundamentally infeasible. This is unfortunate, as automatic proof methods for nonlinear real arithmetic are crucially needed in both formalised mathematics and the verification of real-world cyber-physical systems. Consequently, many researchers have proposed fast, sound but incomplete **RCF** proof procedures which are useful in various practical applications. We show how such practically useful, sound but incomplete **RCF** proof methods may be systematically utilised in the context of a complete **RCF** proof method without sacrificing its completeness. In particular, we present an extension of the **RCF** quantifier elimination method Partial CAD (P-CAD) which uses incomplete $\exists$ **RCF** proof procedures to "short-circuit" expensive computations during the lifting phase of P-CAD. We present the theoretical framework and experimental results arising from an implementation in our **RCF** proof tool **RAHD**.

Keywords: decision procedures, nonlinear arithmetic, real closed fields

1 Introduction

Tarski's theorem that the elementary theory of real closed fields (**RCF**) admits effective elimination of quantifiers is one of the longstanding hallmarks of mathematical logic [13]. From this result, the decidability of elementary algebra and geometry readily follow, and a most tantalising situation arises: In principle, every elementary arithmetical conjecture over finite-dimensional real and complex spaces may be decided simply by formalising the conjecture and asking a computer of its truth. So why then do we still not know how many unit hyperspheres may kiss[3] in five dimensions? Is it 41? 42?

The issue is one of complexity. Though decidable, **RCF** is fundamentally infeasible. Due to Davenport-Heinz [5], it is known that there exist families of n-dimensional **RCF** formulas of length $O(n)$ whose only quantifier-free equivalences must contain polynomials of degree $2^{2^{\Omega(n)}}$ and of length $2^{2^{\Omega(n)}}$. Nevertheless, there are countless examples of difficult, high-dimensional **RCF** problems

* This paper reports on work presented in Chapters 7 and 8 of the first author's 2011 University of Edinburgh Ph.D. thesis [9], supervised by the second author.

[3] See, e.g., [11] for background on the kissing problem for n-dimensional hyperspheres.

solved in mathematical and engineering practice. What is the disconnect? (1) **RCF** problems solved in practice are most often solved using an ad hoc combination of methods, *not* by a general decision method. (2) **RCF** problems arising in practice commonly have structural properties dictated by the application domain from which they originated. Such structural properties can often be exploited making such problems more amenable to analysis and pushing them within the reaches of restricted, more efficient variants of known decision methods.

With this in mind, many researchers have in recent years proposed fast, sound but *incomplete* **RCF** proof procedures which have nevertheless been of substantial practical use [1,7,14,10,12,6,4]. This is especially true for formal methods, where improved automated **RCF** proof methods are needed in the formal verification of cyber-physical systems. In these cases, as the **RCF** problems to be analysed are usually machine-generated (and incomprehensibly large), incomplete proof procedures can go a long way. For example, there is no denying the fact that applying a full quantifier elimination algorithm to decide the falsity of a formula such as $\exists x_1, \ldots, x_{100} \in \mathbb{R}\ (x_1 * x_1 + \ldots + x_{100} * x_{100} < 0)$ is an obvious misappropriation of resources. While such an example may seem contrived, consider the fact that when an **RCF** proof method is used in formal verification efforts, it is often fed huge collections of machine-generated formulas which may be (un)satisfiable for extremely simple reasons. Ideally, one would like to be able to use fast, sound but incomplete proof procedures as much as possible, falling back on the far more computationally expensive complete methods only when necessary. It would be desirable to have a principled manner in which incomplete proof methods could be used to improve the performance of a complete method without sacrificing its completeness.

We present *Abstract Partial Cylindrical Algebraic Decomposition* (AP-CAD), an extension of the **RCF** quantifier elimination procedure partial CAD. In AP-CAD, arbitrary sound but possibly incomplete $\exists$ **RCF** proof procedures may be used to "short-circuit" certain expensive computations during CAD construction. This is done in such a way that the completeness of the combined proof method is guaranteed. We restrict our AP-CAD presentation to the practically useful case of $\exists$ **RCF**. We have implemented AP-CAD within our **RCF** proof tool **RAHD** [9] for the case of full-dimensional cell decompositions and present experiments. **RAHD** contains many **RCF** proof methods and allows users to combine them into their own heuristic **RCF** proof procedures through a *proof strategy language*. This is ideal for AP-CAD, as the proof procedure parameters used by AP-CAD can be formally realised as **RAHD** proof strategies.

2 CAD Preliminaries

For a detailed account of CAD, we refer the reader to [2]. We present only the background on (P-)CAD required to understand AP-CAD for $\exists$ **RCF**. P-CAD is currently the most efficient known general quantifier elimination method

for **RCF**[4]. An important fact is that the complexity of the (P-)CAD decision algorithm is doubly exponential in the dimension (number of variables) of its input formula. Generally, the most expensive phase of the (P-)CAD algorithm is the so-called "lifting phase." Let us fix some notation.

A *semialgebraic set* is a subset of $\mathbb{R}^n$ definable by a quantifier-free formula in the language of ordered rings. A *region* of $\mathbb{R}^n$ is a connected component of $\mathbb{R}^n$. An *algebraic decomposition* of $\mathbb{R}^n$ is a decomposition of $\mathbb{R}^n$ into finitely many semialgebraic regions. A *cylindrical algebraic decomposition* is a special type of algebraic decomposition whose regions are in a sense "well-behaved" with respect to projections onto lower dimensions. A *cell* is a region of a CAD.

Before delving into technical details, let us discuss how we can use a CAD to make $\exists$ **RCF** decisions. From now on, when we say "the polynomials of (an $\exists$ **RCF** formula) φ," we mean the collection of polynomials obtained by zeroing the RHS of every atom in φ through subtracting the RHS from both sides. We assume each such $\exists$ **RCF** formula is in prenex normal form, so that it is an $\exists$-closed boolean combination of *sign conditions*, i.e., of atoms of the form $(p \odot 0)$ with $p \in \mathbb{Z}[x_1, \ldots, x_n]$, $\odot \in \{<, \leq, =, \geq, >\}$. We use $QF(\varphi)$ to mean the quantifier-free matrix of φ.

The key point is that if we have in hand a suitable CAD $C = \{c_1, \ldots, c_m\} \subseteq \mathbb{R}^n$ derived from an $\exists$ **RCF** formula φ, we can decide the truth of φ from the CAD directly. The reason is simple: C will have the property that every polynomial of φ has *constant sign* on each c_i, i.e., given p a polynomial of φ and a c_i a cell, it shall hold that $\forall \boldsymbol{r} \in c_i(p(\boldsymbol{r}) = 0) \vee \forall \boldsymbol{r} \in c_i(p(\boldsymbol{r}) > 0) \vee \forall \boldsymbol{r} \in c_i(p(\boldsymbol{r}) < 0)$. Consequently, $QF(\varphi)$ has constant truth value at every point in a given cell. Thus, to decide φ, we simply substitute a single sample point from each c_i into $QF(\varphi)$ and see if it ever evaluates to **true**. It will evaluate to **true** on at least one sample point if and only if φ is **true** over $\mathbb{R}^n$.

We shall define CAD by induction on dimension[5]. A CAD of $\mathbb{R}$ is a decomposition of $\mathbb{R}$ into finitely many cells $c_i \subseteq \mathbb{R}$ s.t. each c_i is of the form (i) $\{\alpha_1\}$, or (ii) $]\alpha_1, \alpha_2[$, or (iii) $]-\infty, \alpha_1[$ or $]\alpha_1, +\infty[$ for algebraic real numbers α_i. Let $\mathcal{A}$ be a region of $\mathbb{R}^i$. We call $\mathcal{A} \times \mathbb{R}$ the *cylinder over* $\mathcal{A}$ and denote it by $Z(\mathcal{A})$.

Definition 1 (Stack). *Let $f_1, \ldots, f_k \in \mathcal{C}(\mathcal{A}, \mathbb{R})$. That is, f_j is a continuous function from $\mathcal{A}$ to $\mathbb{R}$. Furthermore, suppose that the images of the f_j are ordered over $\mathcal{A}$ s.t. $\forall \alpha \in \mathcal{A}\,(f_j(\alpha) < f_{j+1}(\alpha))$. Then, $f_1, \ldots, f_k$ induce a* stack *S over $\mathcal{A}$, where S is a decomposition of $Z(\mathcal{A})$ into $2k+1$ regions of the following form:*

- $r_1 = \{\langle \alpha, x \rangle \mid \alpha \in \mathcal{A}, x < f_1(\alpha)\}$,
  $r_3 = \{\langle \alpha, x \rangle \mid \alpha \in \mathcal{A}, f_1(\alpha) < x < f_2(\alpha)\}$,
  $\vdots$

[4] See [8] for an explanation as to why P-CAD is also currently the best known general decision method for practical $\exists$ **RCF** problems, despite the fact that $\exists$ **RCF** has a theoretical exponential speed-up over **RCF**.

[5] We shall speak freely of the symbolic manipulation and arithmetic of (irrational) real algebraic numbers. See, e.g., [2] for an algorithmic account.

$r_{2k-1} = \{\langle \alpha, x\rangle \mid \alpha \in \mathcal{A}, f_{k-1}(\alpha) < x < f_k(\alpha)\}$,
$r_{2k+1} = \{\langle \alpha, x\rangle \mid \alpha \in \mathcal{A}, f_k(\alpha) < x\}$,

- $r_2 = \{\langle \alpha, x\rangle \mid \alpha \in \mathcal{A}, x = f_1(\alpha)\}$,

$\vdots$

$r_{2k} = \{\langle \alpha, x\rangle \mid \alpha \in \mathcal{A}, x = f_k(\alpha)\}$.

A CAD of $\mathbb{R}^{i+1}$ will be obtained from a CAD C of $\mathbb{R}^i$ by constructing a stack over every cell in C.

Definition 2 (CAD in $\mathbb{R}^{i+1}$). *An algebraic decomposition C_{i+1} of $\mathbb{R}^{i+1}$ is a CAD iff C_{i+1} is a* union of stacks $C_{i+1} = \bigcup_{j=1}^{k} w_j$, *s.t. the stack w_j is constructed over cell c_j in a CAD $C_i = \{c_1, \ldots, c_k\}$ of $\mathbb{R}^i$.*

The P-invariance property will allow us to use CADs to make $\exists$ **RCF** decisions.

Definition 3 (P-invariance). *Let $P = \{p_1, \ldots, p_k\} \subset \mathbb{Z}[x_1, \ldots, x_n]$ and $\mathcal{A}$ be a region of $\mathbb{R}^n$. Then, we say $\mathcal{A}$ is* P-invariant *iff every member of P has constant sign on $\mathcal{A}$. That is given any $p_i \in P$,*

$$\forall \boldsymbol{r} \in \mathcal{A}(p_i(\boldsymbol{r}) = 0) \quad \vee \quad \forall \boldsymbol{r} \in \mathcal{A}(p_i(\boldsymbol{r}) > 0) \quad \vee \quad \forall \boldsymbol{r} \in \mathcal{A}(p_i(\boldsymbol{r}) < 0).$$

Given a CAD C, we say C is P-invariant iff every cell of C is P-invariant.

2.1 CAD Construction and Evaluation for $\exists$ RCF

The use of CADs for deciding $\exists$ **RCF** sentences will take place in four steps. In what follows, φ is an $\exists$ **RCF** sentence and $P = \{p_1, \ldots, p_k\} \subset \mathbb{Z}[x_1, \ldots, x_n]$ is the collection of polynomials of φ.

Projection The *projection phase* will begin with P and iteratively apply a *projection operator* $Proj$ of the form $Proj : 2^{\mathbb{Z}[x_1,\ldots,x_{i+1}]} \to 2^{\mathbb{Z}[x_1,\ldots,x_i]}$ until a set of polynomials is obtained over $\mathbb{Z}[x_1]$. This process will consist of levels, one for each dimension, s.t. at each level i we will have what is called a *level-i projection set*, $P_i \subset \mathbb{Z}[x_1, \ldots, x_i]$. These level-$i$ projection sets will have a very special property: Namely, if we have a P_i-invariant CAD of $\mathbb{R}^i$, then we can use this CAD to construct a P_{i+1}-invariant CAD of $\mathbb{R}^{i+1}$.

Base The *base phase* consists of computing a P_1-invariant CAD of $\mathbb{R}^1$, implicitly described as a sequence of sample points, one for each cell in the CAD. This can be done by *univariate real root isolation* and basic machinery for arithmetic with real algebraic numbers. Let us suppose we have done this and our sequence of sample points is $s_1 < s_2 < \ldots < s_{2m+1}$.

Lifting The *lifting phase* will take an implicit description of a P_1-invariant CAD of $\mathbb{R}^1$ and progressively transform it into an implicit description of P_n-invariant CAD of $\mathbb{R}^n$. Let $C = \{c_1, \ldots, c_m\}$ be the P_i-invariant CAD for $\mathbb{R}^i$ which we will lift to a P_{i+1}-invariant CAD of $\mathbb{R}^{i+1}$. Let $S = \{s_1, \ldots, s_m\}$ be our set of sample points, one from each cell in C. Then, for each cell c_j, we will use the sample point $s_j \in c_j$ to construct a set of sample points in $\mathbb{R}^{i+1}$ corresponding to a *stack over c_j*:

1. As $s_j \in \mathbb{R}^i$, we have that $s_j = \langle r_1, \ldots, r_i \rangle$ for some $r_1, \ldots, r_i \in \mathbb{R}$.
2. Let $P_{i+1}[s_j]$ denote $P_i[x_1 \mapsto r_1, x_2 \mapsto r_2, \ldots, x_i \mapsto r_i]$. Then $P_{i+1}[s_j] \subset \mathbb{Z}[x_{i+1}]$ is a univariate family of polynomials.
3. Using the same process as we did in the base phase, compute a $P_{i+1}[s_j]$-invariant CAD of $\mathbb{R}^1$. Let this CAD be represented by a sequence of sample points $t_1 < t_2 < \ldots < t_{2v+1} \in \mathbb{R}$.
4. Then, the *stack over* c_j will be represented by the set of $2v+1$ sample points obtained by appending each t_j to the lower-dimensional sample point s_j. That is, our stack over c_j will be represented by the following sequence of sample points $z_1, \ldots, z_{2v+1}$ in $\mathbb{R}^{i+1}$: $z_1 = \langle r_1, \ldots, r_i, t_1 \rangle$, $z_2 = \langle r_1, \ldots, r_i, t_2 \rangle$, …, $z_{2v+1} = \langle r_1, \ldots, r_i, t_{2v+1} \rangle$.

In the above construction, we call the cell c_j (or the sample point representing it, s_j) the *parent* of the stack $\{z_1, \ldots, z_{2v+1}\}$.

Evaluation Let $S = \{s_1, \ldots, s_m\} \subset \mathbb{R}^n$ be our final set of sample points. Return the boolean value $\bigvee_{\boldsymbol{r} \in S} QF(\varphi)[\boldsymbol{r}]$.

2.2 Partial CAD

Let us now sketch the idea of partial CAD, due to Collins and Hong [3]. As it stands, the CAD construction algorithm will build a P-invariant CAD induced by the polynomials P of an $\exists$ **RCF** formula φ without paying any attention to the logical content of the formula itself. But, when performing lifting, i.e., constructing a stack of regions of $\mathbb{R}^{i+1}$ over a lower-dimensional cell $c_j \subset \mathbb{R}^i$, we may be easily able to see — simply by substitution and evaluation — that the formula $QF(\varphi)$ could *never* be satisfied over c_j. For instance, let $QF(\varphi) = \left((x_4^4 + x_3 x_2^3 + 3x_1 > 2x_1^4) \ \wedge \ (x_1^2 > x_2 + x_3)\right)$. If c_j is a cell in a P_3-invariant CAD of $\mathbb{R}^3$ represented by the sample point $s_j = \langle 0, 1, 5 \rangle$, then we can see $QF(\varphi)$ will *never* be satisfied over a cell in a stack which is a child of c_j. Thus, we need not lift over c_j and can eliminate it.

This is the idea behind *partial CAD* when applied to $\exists$ **RCF** formulas: Before lifting over a cell in a CAD of $\mathbb{R}^i$, check if there are any atoms in your formula only involving the variables $x_1, \ldots, x_i$. If so, then perform *partial* evaluation of your formula by evaluating those atoms upon your sample point in $\mathbb{R}^i$, and then use simple propositional reasoning to try to deduce the truth of your formula. This can also allow us to find a *satisfying* assignment for the variables in $QF(\varphi)$ without constructing a whole CAD. For instance, let $QF(\varphi) = \left((x_4^4 + x_3 x_2^3 + 3x_1 > 2x_1^4) \ \vee \ (x_1^2 < x_2)\right)$. If c_j is a cell in a P_2-invariant CAD of $\mathbb{R}^2$ represented by the sample point $s_j = \langle -1, 2 \rangle$, then we can see immediately by substitution that $QF(\varphi)$ is satisfiable over $\mathbb{R}^4$. As a witness to this satisfiability, we may return $\langle -1, 2, r_3, r_4 \rangle$ where $r_3, r_4 \in \mathbb{R}$ are arbitrary.

3 Abstract Partial CAD

From a high level of abstraction, we can see partial CAD to be normal CAD augmented with three pieces of algorithmic data:

1. A strategy for selecting lower-dimensional cells to use for evaluating lower-dimensional atoms in our input formula,
2. An algorithm which when given a cell c_j will construct a formula $F(c_j)$ which, if it both has a truth value and is decided, can be used to tell (i) if the cell c_j can be thrown away (i.e., $F(c_j)$ is decided to be **false**), or (ii) if a satisfying assignment for our formula can be extracted already from a lower-dimensional cell (i.e., $F(c_j)$ is decided to be **true**),
3. A proof procedure which will be used to decide the formulas $F(c_j)$ generated by the algorithm above.

In fact, in their original paper on partial CAD, Collins and Hong make the point that different cell selection strategies could be used and even implement and experiment with a number of them[6]. For partial CAD restricted to $\exists$ **RCF**, these three pieces of algorithmic data described above would be:

1. Select cells $c_i \in C$ in some specified enumeration order (specified by s): $c_{s(1)}, c_{s(2)}, c_{s(3)}, \ldots$.
2. Given a cell c_j represented by a sample point $s_j = \langle r_1, \ldots, r_i \rangle \in \mathbb{R}^i$, the formula $F(c_j)$ will be constructed from our original $\exists$ **RCF** formula φ by the following process:
   (a) Let φ' be $QF(\varphi)$ augmented by instantiating x_1 with r_1, x_2 with r_2, ...
   (b) Evaluate all variable-free atoms in φ' to obtain a new formula φ''.
   (c) Replace all (unique) variable-containing atoms in φ'' with fresh propositional variables to obtain a new formula $F(c_j)$.
3. Use a propositional logic proof procedure to attempt to decide $F(c_j)$.

If $F(c_j)$ is **false** (i.e., unsatisfiable), cell c_j can be abandoned and we need not lift over it. If $F(c_j)$ is **true** (i.e., tautologous), then we can extract a witness to the truth of φ from the sample point s_j. Otherwise, we lift over c_j. These three pieces of data give us the widely-used partial CAD of Collins and Hong. But, from this point of view, we see that there are *many other choices* we could make.

3.1 Stages, Theatres and Lifting

The fundamental notion of AP-CAD will be that of an *stage*[7]. Let $\mathcal{L}_{\exists OR}$ be the fragment of the language of ordered rings consisting of purely $\exists$ prenex sentences.

Definition 4 (Stage). *A* stage $\mathfrak{A} = \langle \langle \mathbb{S}, w \rangle, \mathbb{F}, \mathbb{P} \rangle$ *will be given by three pieces of algorithmic data. We describe a stage by how it acts in the context of a fixed (but arbitrary) i-dimensional space $\mathbb{R}^i$. These data are as follows:*

[6] For Collins and Hong, a cell selection strategy selects *single* cells in some specified order. In Abstract Partial CAD, cell selection strategies will select *sets* of cells in some specified order and $\exists$ **RCF** proof procedures will be applied to see if every cell in a selected set of cells may be eliminated.

[7] The intended connotation is of a *stage* in a *theatre*.

1. *A* cell selection strategy *for selecting* subsets *of* C_i *for analysis (we call such a subset a "selection of cells"),*
2. *A* formula construction strategy *for constructing an* $\exists$ ***RCF*** *formula whose truth value will correspond to the relevance of a selection of cells (we call such a formula a "cell selection relevance formula"),*
3. *An* $\exists$ ***RCF*** *proof procedure used to (attempt to) decide the truth or falsity of a cell selection relevance formula.*

In the context of CAD construction, sample points will be eliminated when their corresponding cells are deemed to be irrelevant to the $\exists$ ***RCF*** *formula inducing the CAD. This removal might then result in a set of sample points for which the cell selection function behaves differently than it did initially. This motivates the* convergence axiom *for covering width functions, so that these dynamics do not result in a non-terminating CAD-based decision algorithm employing the stage machinery. In what follows, let* $R_i = \{s \subset \mathbb{R}^i \mid |s| < \omega\}$.

1. *A* cell selection strategy *consists of two components:*
   (a) *A* covering width function $w : R_i \to \mathbb{N}$,
   (b) *A* cell selection function $\mathbb{S} : R_i \times \mathbb{N} \to R_i$ *obeying for all* $S_i \in R_i$ *and all* $j \in \{1, \ldots, w(S_i)\}$ *the* containment axiom*:* $\mathbb{S}(S_i, j) \subset S_i$.
2. *A* formula construction strategy *is a function* $\mathbb{F} : \mathcal{L}_{\exists OR} \times R_i \to \mathcal{L}_{\exists OR}$ *obeying certain* relevance judgment axioms*. To describe these axioms, we need the context of a fixed (but arbitrary)* $\exists$ ***RCF*** *formula and an associated* P_i*-invariant CAD of* $\mathbb{R}^i$*. Let* φ *be an* $\exists$ ***RCF*** *formula with polynomials* $P \subset \mathbb{Z}[x_1, \ldots, x_n]$ *and let* $P_n, \ldots, P_1$ *be a sequence of level-*$(n, \ldots, 1)$ *projection sets rooted in* P *(recall* $P_n = P$*). Let* $C_i = \{c_1, \ldots, c_m\}$ *be a* P_i*-invariant CAD of* $\mathbb{R}^i$ *with* S_i *a set of sample points drawn from a subset of the cells in* C_i*. If we are given a set of sample points* $\{s_{a_1}, \ldots s_{a_v}\} \subseteq S_i$*, then* $\triangle(\{s_{a_1}, \ldots s_{a_v}\})$ *will denote the set of cells from which the sample points* s_{a_j} *are drawn. Then, for each set of sample points* S_i *and each* $j \in \{1, \ldots, w(S_i)\}$ *the following* relevance judgment axioms *must hold:* $[(\boldsymbol{RCF} \models \neg\mathbb{F}(\varphi, \mathbb{S}(S_i, j))) \implies \mathcal{N}(\varphi, \mathbb{S}(S_i, j))]$*, and* $[(\boldsymbol{RCF} \models \mathbb{F}(\varphi, \mathbb{S}(S_i, j))) \implies \boldsymbol{RCF} \models \varphi]$*, where* $\mathcal{N}(\varphi, \{s_{a_1}, \ldots s_{a_v}\})$ *means that no child (at any ancestral depth, i.e., in a* P_{i+1}*-invariant CAD of* $\mathbb{R}^{i+1}$*, in a* P_{i+2}*-invariant CAD of* $\mathbb{R}^{i+2}, \ldots$*, in a* P_n*-invariant CAD of* $\mathbb{R}^n$*) of any cell in the set* $\Delta(\{s_{a_1}, \ldots, s_{a_v}\})$ *will satisfy* $QF(\varphi)$*.*
3. *An* $\exists$ ***RCF*** proof procedure *is a function*

$$\mathbb{P} : \mathcal{L}_{\exists OR} \to \left(\{\boldsymbol{true}, \boldsymbol{false}, \boldsymbol{unknown}\} \cup \bigcup_{j \in \mathbb{N}^+} \mathbb{R}^j \right)$$

obeying the soundness axioms*:* $((\mathbb{P}(\psi) = \boldsymbol{true}) \implies \boldsymbol{RCF} \models \psi)$, $((\mathbb{P}(\psi) = \boldsymbol{false}) \implies \boldsymbol{RCF} \models \neg\psi)$, $((\mathbb{P}(\psi) \in \mathbb{R}^j) \implies \boldsymbol{RCF} \models QF(\psi)[\mathbb{P}(\psi)])$ *for arbitrary* $\psi \in \mathcal{L}_{\exists OR}$ *and with* $QF(\psi)[\mathbb{P}(\psi)]$ *in the final axiom being the substitution of the* j*-vector* $\mathbb{P}(\psi)$ *(or an arbitrary extension of it to the dimension of the polynomials appearing in* ψ*) into* ψ*, resulting in a variable-free formula. In this case, we call* $\mathbb{P}(\psi)$ *a* witness *to the truth of* ψ*.*

We will want to have the freedom to give our AP-CAD algorithm a *sequence* of stages, one for each dimension $1, \ldots, n$. The intuition is as follows: Stages are introduced so that one can present a *strategy* to an underlying CAD decision algorithm which will prescribe a method for the algorithm to recognise when it can short-circuit certain expensive computations. If we can abandon a cell at a low-dimension, for instance at the base phase or when beginning to lift over cells of $\mathbb{R}^2$, then this can potentially give us *hyper-exponential* savings later. Thus, it makes sense to arrange stages $\mathfrak{A}_1, \mathfrak{A}_2, \ldots \mathfrak{A}_n$ so that stage $\mathfrak{A}_1$ works *hardest* to make relevance judgments about cells. For if $\mathfrak{A}_1$ causes us to throw away cell $c_j \subset \mathbb{R}^1$, then this could lead to huge savings later. Then, stage $\mathfrak{A}_2$ might still work hard but a bit *less* hard, and so on. This collection of stages gives rise to the notion of an *n-theatre*. In what follows, let Θ be the set of all stages.

Definition 5 (Theatre). *An* n-theatre $\mathbb{T}$ *is a function* $\mathbb{T} : \{1, \ldots, n\} \to \Theta$.

Stage i in a theatre will be used to make judgments about cells in a P_i-invariant (partial) CAD of $\mathbb{R}^i$ (i.e., at level i). Let us describe a decision method we will use for deciding $\exists$ **RCF** sentences in the framework of AP-CAD.

Algorithm 1 (AP-CAD with Theatrical Lifting) *Suppose we are given an* $\exists$ ***RCF*** *sentence* φ *with polynomials* $P \subset \mathbb{Z}[x_1, \ldots, x_n]$*, and an* n*-theatre* $\mathbb{T}$*.*

1. ***Projection*** *As with standard P-CAD, compute a sequence of projection sets* $P_1, \ldots, P_n$*.*
2. ***Base*** *As with standard P-CAD, compute a* P_1*-invariant CAD of* $\mathbb{R}^1$*,* $C_1 = \{c_1, \ldots, c_{2m+1}\}$ *represented by sample points* $S_1 = \{s_1, \ldots, s_{2m+1}\}$*. Set the current dimension* $i := 1$*.*
3. ***Lifting*** *Let* $\mathbb{T}(i) = \mathfrak{A}_i = \langle\langle \mathbb{S}_i, w_i \rangle, \mathbb{F}_i, \mathbb{P}_i \rangle$ *be the stage for dimension* i*, and* S_i *the set of sample points for the* P_i*-invariant (partial) CAD of* $\mathbb{R}^i$ *over which we need to lift.*
   (a) *Let* $U := w_i(S_i)$ *and let* $j := 1$*.*
   (b) ***While*** $j \leq U$ ***do***
      i. *Let* $\{s_{a_1}, \ldots, s_{a_v}\} := \mathbb{S}_i(S_i, j)$*.*
      ii. *Let* $\chi := \mathbb{P}_i(\mathbb{F}_i(\{s_{a_1}, \ldots, s_{a_v}\}))$*.*
      iii. *If* $\chi =$ ***true****, then* ***return true****.*
      iv. *If* $\chi = \langle x_1, \ldots, x_w \rangle \in \mathbb{R}^w$ *for some* $w \leq n$*, then*
         A. *Fix an* n*-dim. extension of* χ*, e.g.,* $\boldsymbol{r} = \langle x_1, \ldots, x_w, \mathbf{0} \rangle \in \mathbb{R}^n$*.*
         B. *Evaluate* $QF(\psi)[\boldsymbol{r}]$ *and set* $R \in \{\boldsymbol{true}, \boldsymbol{false}\}$ *to this result.*
         C. *If* $R =$ ***true****, then* ***return r*** *as a witness to the truth of* φ*.*
         D. *If* $R =$ ***false****, then* ***return true***[8]*.*

[8] This is perhaps the one counter-intuitive part of the algorithm. Note, however, that this is actually correct: By the combination of the second *relevance judgment axiom* for $\mathbb{F}_i$ and the *soundness axioms* for $\mathbb{P}_i$, the fact that $\mathbf{RCF} \models \mathbb{F}_i(\mathbb{S}_i(S_i, j))$ means that φ is **true**. It's just that the witness $\mathbb{P}_i$ computed for the truth of $\mathbb{F}_i(\mathbb{S}_i(S_i, j))$ might fail to be a witness for φ. In this case, we simply know φ is true without knowing a witness for it.

v. *If* $\chi =$ ***false****, then set* $S'_i := S_i \setminus \{s_{a_1}, \ldots, s_{a_v}\}$*, else set* $S'_i := S_i$*.*
vi. *If* $S'_i = \emptyset$ *then* ***return false****.*
vii. *If* $S'_i = S_i$ *then set* $j := j + 1$*.*
viii. *If* $S'_i \subset S_i$ *then*
A. *Set* $S_i := S'_i$*.*
B. *Set* $U := w_i(S_i)$*.*
C. *Set* $j := 1$*.*

(c) *Now,* $S_i = \{t_1, \ldots, t_u\}$ *contains sample points corresponding to the cells we have not deemed to be irrelevant. We need to lift over them.*
i. *Let* $S_{i+1} := \emptyset$*.*
ii. ***For*** j ***from*** 1 ***to*** u ***do***
A. *Substitute the components of* t_j *in for the variables* $x_1, \ldots, x_i$ *in* P_{i+1} *to obtain a univariate family* $P_{i+1}[t_j] \subset \mathbb{Z}[x_{i+1}]$*.*
B. *Compute a* $P_{i+1}[t_j]$*-invariant CAD of* $\mathbb{R}^1$*, represented by sample points* K_j*.*
C. *Set* $S_{i+1} := S_{i+1} \cup K_j$*.*

(d) *Increase the current dimension by setting* $i := i + 1$*.*
(e) *If* $i = n$ *then lifting is done and we may proceed to the* evaluation phase*.*
(f) *If* $i < n$ *then we loop and begin the lifting process again, but now with the set of sample points* S_{i+1}*.*

4. ***Evaluation*** *Return the boolean value* $\bigvee_{\boldsymbol{r} \in S_n} QF(\varphi)[\boldsymbol{r}]$*.*

(See **Appendix A** for a proof of correctness of the above algorithm.)

4 Experimental Results

As an experiment (explicated in **Appendices B** and **C**), we built a concrete AP-CAD theatre combining interval constraint reasoning with standard partial CAD [9]. As CAD performance is strongly dependent on the number of cells retained at each dimension, we compared this for three CAD variants: (i) CAD, (ii) standard P-CAD, and (iii) AP-CAD, w.r.t. an $\exists$ **RCF** sentence φ s.t.

$$QF(\varphi) = \left[\begin{array}{r} (x_1x_4 + x_2x_4 + x_3x_2 < 0) \wedge \ (x_2 > 0) \ \wedge \ (x_3 > 0) \ \wedge \ (x_4 > 0) \\ \wedge \ (x_3x_4 - x_4^2 + x_3^2 + 1 < 0) \end{array} \right].$$

As $QF(\varphi)$ involves only strict inequalities, we appeal to a theorem of McCallum allowing us to only consider full-dimensional cells (selecting rational sample points), and compare the methods w.r.t. this CAD variant [9]. We observe that the AP-CAD method retains significantly less cells than the other methods. This is supported by experiments we have done with other $\exists$ **RCF** formulas (cf. **Appendix C**). In all cases, the cost of AP-CAD theatre execution measured $< 0.01\%$ of the total CPU time, indicating that there is much positive impact to be made by using incomplete **RCF** proof procedures to enhance the performance of CAD-based decision methods. The cell retainment counts are as follows:

	CAD	P-CAD	AP-CAD
$\mathbb{Q}^1$	2	2	1
$\mathbb{Q}^2$	14	7	5
$\mathbb{Q}^3$	40	10	7
$\mathbb{Q}^4$	200	50	35

5 Conclusion

AP-CAD allows strategic algorithmic data to be used to "short-circuit" expensive computations during the *lifting phase* of a CAD-based decision algorithm. This provides a principled approach for utilising fast, sound but possibly incomplete $\exists$ **RCF** proof procedures to enhance a *complete* decision method without threatening its completeness. We see many ways this work might be extended. It would be very interesting to work out similar machinery to be used during the *projection phase* of P-CAD. For this work to bear serious practical fruit, many more AP-CAD stages should be constructed and experimented with heavily.

References

1. J. Avigad and H. Friedman. Combining Decision Procedures for the Reals. In *Logical Methods in Computer Science*, 2006.
2. S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*. Springer-Verlag, Secaucus, NJ, USA, 2006.
3. G. E. Collins and H. Hong. Partial Cylindrical Algebraic Decomposition for Quantifier Elimination. *J.Sym.Comp.*, 12(3):299–328, 1991.
4. M. Daumas, D. Lester, and C. Muñoz. Verified Real Number Calculations: A Library for Interval Arithmetic. *IEEE Trans. Comp.*, 58(2):226–237, 2009.
5. J. H. Davenport and J. Heintz. Real Quantifier Elimination is Doubly Exponential. *J. Symb. Comput.*, 5:29–35, 1988.
6. S. Gao, M. Ganai, F. Ivancic, A. Gupta, S. Sankaranarayanan, and E. Clarke. Integrating ICP and LRA solvers for deciding nonlinear real arithmetic problems. In *FMCAD, 2010*, pages 81–89, 2010.
7. J. Harrison. Verifying Nonlinear Real Formulas via Sums of Squares. In *TPHOLs'07*, pages 102–118, Berlin, Heidelberg, 2007. Springer-Verlag.
8. H. Hong. Comparison of Several Decision Algorithms for the Existential Theory of the Reals. Technical report, RISC, 1991.
9. G. O. Passmore. *Combined Decision Procedures for Nonlinear Arithmetics, Real and Complex*. PhD thesis, University of Edinburgh, 2011.
10. G. O. Passmore and P. B. Jackson. Combined Decision Techniques for the Existential Theory of the Reals. In *Calculemus'09*, 2009.
11. F. Pfender and G. M. Ziegler. Kissing Numbers, Sphere Packings, and Some Unexpected Proofs. *Notices of the A.M.S.*, 51:873–883, 2004.
12. A. Platzer, J.-D. Quesel, and P. Rümmer. Real World Verification. In *CADE-22: Proceedings of the 22nd International Conference on Automated Deduction*, pages 485–501, Berlin, Heidelberg, 2009. Springer-Verlag.
13. A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. RAND Corporation, 1948.
14. A. Tiwari. An Algebraic Approach for the Unsatisfiability of Nonlinear Constraints. In *CSL 2005*, volume 3634 of *LNCS*, pages 248–262. Springer, 2005.

A Main Correctness Theorem

Theorem 2 (Correctness of AP-CAD with Theatrical Lifting) *Let us prove the correctness of* ***Algorithm 1****. This will be straight-forward given the correctness of the classical CAD-based decision algorithm outlined previously, which we accept as given.*

Proof. There are two essential differences between this AP-CAD algorithm and the classical one. These both take place during lifting. In ***Algorithm 1****, we may*

- *discard a collection of cells if they are deemed to be irrelevant, and*
- *quit CAD construction altogether and return either* ***true*** *or a witness to the truth of our input formula φ, or return* ***false*** *in the case that all cells have been discarded.*

If $\{s_{a_1}, \ldots, s_{a_v}\} \subset \mathbb{R}^i$ is a set of sample points for a P_i-invariant partial CAD of $\mathbb{R}^i$, we will say that $\{s_{a_1}, \ldots, s_{a_v}\}$ ***respects the truth of*** *φ to mean that there is some n-dimensional child of a sample point in $\{s_{a_1}, \ldots, s_{a_v}\}$ satisfying $QF(\varphi)$ iff φ is* ***true****.*

We will proceed by induction, assuming that the algorithm has constructed a set of sample points $\{s_{a_1}, \ldots, s_{a_v}\}$ for a P_i-invariant partial CAD of $\mathbb{R}^i$ which respects the truth of φ. The base case is verified by noting that the base phase *of the algorithm constructs a full set of sample points for a P_1-invariant CAD of $\mathbb{R}^1$ which trivially respects the truth of φ.*

Let us first observe that if we discard a collection of cells because they have been deemed to be irrelevant, then we have not affected the soundness of the decision algorithm.

Cells of a P_i-invariant partial CAD of $\mathbb{R}^i$ will only be deemed to be irrelevant when an stage $\mathfrak{A}_i$ *indicates this is the case. The key line in the algorithm is 3(b)v, where $\chi = \mathbb{P}_i(\mathbb{F}_i(\varphi, \{s_{a_1}, \ldots, s_{a_v}\}))$. For this discarding to have occurred, we must have $\chi =$* ***false****. By the* second soundness axiom *for $\mathbb{P}_i$, this means*

$$\boldsymbol{RCF} \models \neg \mathbb{F}_i(\varphi, \{s_{a_1}, \ldots, s_{a_v}\}).$$

By the first relevance judgment axiom *for $\mathbb{F}_i$, this means that $\mathcal{N}(\varphi, \{s_{a_1}, \ldots, s_{a_v}\})$ must hold. Recall that $\mathcal{N}(\varphi, \{s_{a_1}, \ldots s_{a_v}\})$ means that no child (at any ancestral depth, i.e., in a P_{i+1}-invariant CAD of $\mathbb{R}^{i+1}$, in a P_{i+2}-invariant CAD of $\mathbb{R}^{i+2}, \ldots$, in a P_n-invariant CAD of $\mathbb{R}^n$) of any cell in the set $\Delta(\{s_{a_1}, \ldots, s_{a_v}\})$ of cells corresponding to the sample points $\{s_{a_1}, \ldots, s_{a_v}\}$ will satisfy $QF(\varphi)$. Thus, by our induction hypothesis, removing the cells from our analysis does not affect the soundness of the decision algorithm. In particular, if we have removed all cells, this means that* no *ancestor of the cells at our current dimension can satisfy $QF(\varphi)$. By our induction hypothesis this means that there exists no n-dimensional real vector satisfying $QF(\varphi)$, and thus φ is* ***false*** *as the algorithm will report via line 3(b)vi.*

Let us now turn to the second difference: ***Algorithm 1*** *may quit CAD construction altogether and return either* ***true*** *or a witness satisfying $QF(\varphi)$.*

In the latter case, a witness is only returned if the algorithm verified, by evaluation, that the witness satisfies $QF(\varphi)$. That this does not affect soundness is apparent.

Let us examine the remaining case, when the algorithm returns simply ***true*** *during lifting. The first place this occurs is on line 3(b)iii. This happens when* $\mathbb{P}_i(\mathbb{F}_i(\varphi, \{s_{a_1}, \ldots, s_{a_v}\}))$ *is equal to* ***true****. By the* first soundness axiom *for* $\mathbb{P}_i$*, this means*

$$\boldsymbol{RCF} \models \mathbb{F}_i(\varphi, \{s_{a_1}, \ldots, s_{a_v}\}).$$

By the second relevance judgment axiom *for* $\mathbb{F}_i$*, it then follows that* φ *is in fact* ***true*** *over* ***RCF*** *and so the soundness of the algorithm is not affected.*

Finally, let us consider the second scenario in which this could occur, line 3(b)ivD. In this case, $\mathbb{P}_i(\mathbb{F}_i(\varphi, \{s_{a_1}, \ldots, s_{a_v}\})) \in \mathbb{R}^j$ *for some* $j \in \mathbb{N}$*. By the* third soundness axiom *for* $\mathbb{P}_i$*, this means that*

$$\boldsymbol{RCF} \models QF(\mathbb{F}_i(\varphi, \{s_{a_1}, \ldots, s_{a_v}\}))[\mathbb{P}_i(\mathbb{F}_i(\varphi, \{s_{a_1}, \ldots, s_{a_v}\}))].$$

But this implies that

$$\boldsymbol{RCF} \models \mathbb{F}_i(\varphi, \{s_{a_1}, \ldots, s_{a_v}\})).$$

So, as in the last case, by the second relevance judgment axiom *for* $\mathbb{F}_i$*, this means that* φ *is in fact* ***true****.*

Finally, a word on termination of the while loop (cf. line 3b): Consider a pass of the loop. If any sample points in S_i *are discarded, then* $|S_i|$ *is reduced. If no sample points in* S_i *are discarded, then* U *remains constant and* j *is incremented by* 1*. Thus, the lexicographic product measure* $\mu = \langle |S_i|, U - j + 1 \rangle$ *is always decreased along the ordinal* ω^2*. If ever* $|S_i|$ *is reduced to* 0*, then line 3(b)vi guarantees termination. Combining this with the fact that the loop termination condition is* $(j > U)$*, it follows by the well-foundedness of* ω^2 *that the loop must terminate.*

Thus, by the correctness of the classical CAD-based decision algorithm, it follows by induction that ***Algorithm 1*** *is sound and terminating.*

B Definition of our AP-CAD Theatre

In defining this theatre, it will be useful to allow our functions to work explicitly over *lists* of sample points as opposed to sets of sample points. To do so, we use the maps

$$StoL : R_i \to Lists(R_i)$$

and

$$LtoS : Lists(R_i) \to R_i.$$

StoL will map a set of sample points to a *sorted* representation of the set as a list, and *LtoS* will map a list of sample points to its underlying set. We use the lexicographic product order of the normal ordering $<$ on $\mathbb{R}$ to order the

sample points. If l is a list, then $|l|$ will be the length of the list. If l is a list and $0 \leq m \leq n \leq |l|$, then $subseq(l, m, n)$ will be the subsequence of l of the form[9] $\langle l(m), \ldots, l(n-1)\rangle$.

We build now a stage for our theatre.

cell selection function $\mathbb{S}(s, n) = LtoS(\mathbb{S}_{Lists}(StoL(s), n))$ where

$$\mathbb{S}_{Lists}(l, n) = \begin{cases} l & \text{if } n \leq 1, \\ \lfloor S_{Lists}(l, k) \rfloor & \text{if } n = 2k, \\ \lceil S_{Lists}(l, k) \rceil & \text{if } n = 2k+1, \end{cases}$$

and

$$\lfloor l \rfloor = \begin{cases} subseq(l, 0, k) & \text{if } |l| = 2k, \\ subseq(l, 0, k+1) & \text{if } |l| = 2k+1, \end{cases}$$

and

$$\lceil l \rceil = \begin{cases} subseq(l, k, |l|) & \text{if } |l| = 2k, \\ subseq(l, k+1, |l|) & \text{if } |l| = 2k+1. \end{cases}$$

Let us explain these functions in words. The function $\lfloor l \rfloor$ returns the first half of the list l if $|l|$ is even, and returns the first $k+1$ elements of l if $|l| = 2k+1$. The function $\lceil l \rceil$ returns the second half of the list l if $|l|$ is even, and returns the final k elements of l if $|l| = 2k+1$. In this way, we always have that the concatenation of $\lfloor l \rfloor$ and $\lceil l \rceil$ is l itself. These two functions are used to "bisect" the list l by the function $\mathbb{S}_{Lists}$, regardless of whether or not $|l|$ is even or odd.

The function $\mathbb{S}_{Lists}(l, n)$ computes subsequences of the list l in a "divide and conquer" fashion, with the parameter n specifying which subsequence should be computed. It is best understood as representing an enumeration of subsequences of l which have been situated in a binary tree. To illustrate a concrete example, let $l = \langle a_1, \ldots, a_7\rangle$. Then, we have

$$\begin{aligned}
&\mathbb{S}_{Lists}(l, 1) = l = \langle a_1, \ldots, a_7\rangle, \\
&\mathbb{S}_{Lists}(l, 2) = \lfloor \langle a_1, \ldots, a_7\rangle \rfloor = \langle a_1, \ldots, a_4\rangle, \\
&\mathbb{S}_{Lists}(l, 3) = \lceil \langle a_1, \ldots, a_7\rangle \rceil = \langle a_5, \ldots, a_7\rangle, \\
&\mathbb{S}_{Lists}(l, 4) = \lfloor \lfloor \langle a_1, \ldots, a_7\rangle \rfloor \rfloor = \lfloor \langle a_1, \ldots, a_4\rangle \rfloor = \langle a_1, a_2\rangle, \\
&\mathbb{S}_{Lists}(l, 5) = \lceil \lfloor \langle a_1, \ldots, a_7\rangle \rfloor \rceil = \lceil \langle a_1, \ldots, a_4\rangle \rceil = \langle a_3, a_4\rangle, \\
&\mathbb{S}_{Lists}(l, 6) = \lfloor \lceil \langle a_1, \ldots, a_7\rangle \rceil \rfloor = \lfloor \langle a_5, \ldots, a_7\rangle \rfloor = \langle a_5, a_6\rangle, \\
&\mathbb{S}_{Lists}(l, 7) = \lceil \lceil \langle a_1, \ldots, a_7\rangle \rceil \rceil = \lceil \langle a_5, \ldots, a_7\rangle \rceil = \langle a_7\rangle.
\end{aligned}$$

The cell selection function $\mathbb{S}(s, n)$ then maps s to an underlying sorted list representation $StoL(s)$ and uses $\mathbb{S}_{Lists}$ to compute the nth subsequence of

[9] This perhaps strange way of indexing list subsequences is used so that our description matches our actual implemention, as this is how Common Lisp does list subsequencing via the `subseq` function. For example, `subseq(`$\langle a, b, c\rangle$`, 0, 2) =` $\langle a, b\rangle$ and `subseq(`$\langle a, b, c\rangle$`, 0, 0) = nil`, the empty list.

$StoL(s)$ with respect to the "divide and conquer" enumeration order given above.

covering width function We will use a constant covering width function w of the form

$$w(s) = 3.$$

Given a collection of sample points s, this covering width will cause the AP-CAD lifting algorithm (cf. **Algorithm 1**) to attempt to eliminate the cell selections $\mathbb{S}(s, 1)$ through $\mathbb{S}(s, 3)$. (This is quite a "shallow" depth for a "divide and conquer" strategy. Nevertheless, it will be useful for keeping our explicit examples small enough to discuss in detail and can be easily changed if one wishes to experiment with variations of it.)

formula construction function Our formula construction function $\mathbb{F} : \mathcal{L}_{\exists OR} \times R_i \to \mathcal{L}_{\exists OR}$ will accept an $\exists$ **RCF** formula φ and a set of i-dimensional sample points s and work as follows:

1. Let $min_j(s)$ be the minimal value ever appearing as coordinate j in a sample point in s. To be precise,

$$min_j(s) = min\{\pi_j(x) \mid x \in s\},$$

where π_j projects a sample point $x \in \mathbb{R}^i$ onto its jth coordinate.

2. Similarly, let $max_j(s)$ be s.t.

$$max_j(s) = max\{\pi_j(x) \mid x \in s\}.$$

3. Then,

$$\mathbb{F}(\varphi, s) = \exists \boldsymbol{x} \left[\left(\bigwedge_{j=1}^{i} x_j \geq min_j(s) \ \wedge \ x_j \leq max_j(s) \right) \ \wedge \ QF(\varphi) \right].$$

$\exists$ RCF proof procedure

We use an $\exists$ **RCF** procedure — expressed as a **RAHD** proof strategy — which performs simple formula simplification, saturation of linear bounds on variables, followed by interval constraint propagation (cf. [9] for more information on CMFs):

```
[ simp-zrhs; run stable-simp; satur-lin;
  interval-cp(max-contractions := 30)].
```

RAHD's execution of this proof strategy then gives rise to our AP-CAD stage's $\exists$ **RCF** proof procedure $\mathbb{P}$.

Lemma 1. *$\langle\langle \mathbb{S}, w\rangle, \mathbb{F}, \mathbb{P}\rangle$ as defined above is an AP-CAD stage.*

*Proof. As **RAHD** guarantees that the execution of its proof strategies correspond to proper AP-CAD $\exists$ **RCF** proof procedures, the only non-trivial property*

to verify is that our formula construction function $\mathbb{F}$ *satisfies the* relevance judgment *axioms. Let* φ *be an* $\mathcal{L}_{\exists OR}$ *formula in* $x_1, \ldots, x_n$ *and* $s \subset \mathbb{R}^i$ *a finite set of* i*-dimensional sample points* $(1 \leq i \leq n)$*.*

We must verify that

$$\boldsymbol{RCF} \models \neg\mathbb{F}(\varphi, s) \quad \Longrightarrow \quad \mathcal{N}(\varphi, s),$$

and

$$\boldsymbol{RCF} \models \mathbb{F}(\varphi, s) \quad \Longrightarrow \quad \boldsymbol{RCF} \models \varphi,$$

where (restating the property a bit more concretely than its original axiomatisation in ***Section 3.1****):*

$\mathcal{N}(\varphi, s)$ *means that no child (at any ancestral depth, i.e., in a* P_{i+1}*-invariant CAD of* $\mathbb{R}^{i+1}$*, in a* P_{i+2}*-invariant CAD of* $\mathbb{R}^{i+2}, \ldots,$ *in a* P_n*-invariant CAD of* $\mathbb{R}^n$*) of any sample point in* s *will satisfy* $QF(\varphi)$*.*

In the first case, we have that any child of any sample point in s *will satisfy*

$$\left(\bigwedge_{j=1}^{i} x_j \geq min_j(s) \ \wedge \ x_j \leq max_j(s) \right),$$

and so

$$\boldsymbol{RCF} \models \neg\mathbb{F}(\varphi, s) \quad \Longrightarrow \quad \mathcal{N}(\varphi, s)$$

obviously holds. In the second case,

$$\boldsymbol{RCF} \models \mathbb{F}(\varphi, s) \quad \Longrightarrow \quad \boldsymbol{RCF} \models \varphi$$

is immediate.

Finally, again to keep our detailed examples below from becoming too large, we will turn this AP-CAD stage $\langle\langle \mathbb{S}, w\rangle, \mathbb{F}, \mathbb{P}\rangle$ into an AP-CAD theatre $\mathbb{T}$ in a trivial fashion:

$$\mathbb{T}(n) = \langle\langle \mathbb{S}, w\rangle, \mathbb{F}, \mathbb{P}\rangle.$$

That is, the same stage $\langle\langle \mathbb{S}, w\rangle, \mathbb{F}, \mathbb{P}\rangle$ will be used at every dimension during AP-CAD lifting.

C Experiments in Detail

Let us now apply our concrete AP-CAD theatre to some example $\mathcal{L}_{\exists OR}$ formulas and examine its execution. Recall that the decision method proceeds in four steps: projection, base, lifting and evaluation. We examine each step in detail.

Let φ be as follows:

$$\varphi = \left[\begin{array}{r} \exists x_1 \ \exists x_2 \ \exists x_3 \ \exists x_4 \\ (x_1x_4 + x_2x_4 + x_3x_2 < 0) \\ \wedge \ (x_2 > 0) \ \wedge \ (x_3 > 0) \ \wedge \ (x_4 > 0) \\ \wedge \ (x_3x_4 - x_4^2 + x_3^2 + 1 < 0) \end{array} \right].$$

As φ is an $\exists$ **RCF** sentence s.t. $QF(\varphi)$ consists of a conjunction of strict polynomial inequalities, it follows by McCallum's Theorem (cf. [9]) that we may decide φ by only examining full-dimensional cells during partial CAD construction. This allows us avoid irrational algebraic number computations, as full-dimensional cells (i.e., *sectors*) always contain rational points. This also permits us to use the Brown-McCallum projection operator to obtain our CAD projection factor sets, which can lead to much smaller projection sets than those obtained with projection operators valid for general $\exists$ **RCF** formulas.

We will now walk through using AP-CAD to decide φ. It will turn out that φ is in fact **true** over **RCF**, and we will illustrate the process of constructing a witness to its truth. First, we will compute the projection (factor) sets for φ. Then, we will compute the base phase for our level 1 projection set. Finally, we will show how four different variants of CAD operate during the *lifting* phase. These variants are:

1. full-dimensional lifting without eliminating any full-dimensional cells,
2. full-dimensional lifting with standard partial CAD used to eliminate cells,
3. full-dimensional lifting with our AP-CAD theatre used to eliminate cells,
4. full-dimensional lifting with a combination of standard partial CAD and our AP-CAD stage used to eliminate cells.

With each progressive variant, the number of cells eliminated during lifting will change. The final variant will be the best in the sense that it will allow us to eliminate the most cells during partial CAD construction.

Projection Sets We first extract the polynomials of φ, which gives us $P_4 \subset \mathbb{Z}[x_1, x_2, x_3, x_4]$:

$$P_4 = \left\{ \begin{array}{r} x_2, x_3, x_4, x_1x_4 + x_4x_2 + x_3x_2, \\ x_4^2 - x_3x_4 - x_3^2 - 1 \end{array} \right\}.$$

Then, we apply the Brown-McCallum projection operator $BMProj : \mathbb{Z}[x_1, \ldots, x_{i+1}] \to \mathbb{Z}[x_1, \ldots, x_i]$ to obtain $P_3 \subset \mathbb{Z}[x_1, x_2, x_3]$:

$$BMProj(P_4) = P_3 = \left\{ \begin{array}{r} x_2, x_3, -x_1^2x_3^2 - x_1x_3^2x_2 + x_3^2x_2^2 - x_1^2 - 2x_1x_2 - x_2^2, \\ x_3^2 + 1, x_1 + x_2, 5x_3^2 + 4 \end{array} \right\}.$$

We apply $BMProj$ again to obtain $P_2 \subset \mathbb{Z}[x_1, x_2]$:

$$BMProj(P_3) = P_2 = \left\{ -x_1^2 - x_1x_2 + x_2^2, x_1 + 3x_2, x_1 + 2x_2, x_1 + x_2, x_2 \right\}.$$

Finally, we apply $BMProj$ one last time to obtain $P_1 \subset \mathbb{Z}[x_1]$:

$$BMProj(P_2) = P_1 = \left\{ x_1 \right\}.$$

It is worth stating that obtaining a level-1 projection factor set P_1 s.t. $|P_1| = 1$ is quite unusual. Even for this small problem, the size of the respective projection sets can change drastically depending upon the projection variable order used. Nevertheless, the example arising through the use of this projection order is nice as it results in constructions small enough so that a detailed description of the decision process can be given quite compactly.

Base Phase We now compute the base collection of sample points of $\mathbb{R}^1$ induced by our level 1 projection set $P_1 \subset \mathbb{Z}[x_1]$. As our P_1 is rather uncharacteristically a singleton, this is trivial in this particular example. But, let us state what one must do in general for the full-dimensional base phase, so that our walk-through is applicable when P_1 is larger. We will then follow this same sample-point computation process when constructing stacks over cells in the lifting phase. We will build our collection of sample points in the following manner:

1. We process P_1 into a new set $CoPrime(P_1) \subset \mathbb{Z}[x_1]$ so that no two distinct polynomials $p, q \in CoPrime(P_1)$ share a root, while maintaining the invariant that

$$\{r \in \mathbb{R} \mid \exists p \in P_1(p(r) = 0)\} = \{r \in \mathbb{R} \mid \exists p \in CoPrime(P_1)(p(r) = 0)\}.$$

   This can be done using univariate GCD and division. Our P_1 in this example happens to already have this property, so we simply set $CoPrime(P_1) = P_1$.
2. We apply univariate real root isolation to the polynomials in $CoPrime(P_1)$ to obtain a collection of pairwise disjoint compact real intervals $I_1, \ldots, I_k \subset \mathbb{R}$ s.t. every real root of a polynomial $p \in CoPrime(P_1)$ is contained in exactly one interval I_i, and for each interval I_i, there exists only one $p \in CoPrime(P_1)$ s.t. I_i contains a real root of p (here we exploit the fact that no distinct $p, q \in CoPrime(P_1)$ share a root). This gives us a bijection

$$i \ : \ \{r \in \mathbb{R} \mid \exists p \in P_1(p(r) = 0)\} \ \rightarrow \ \{I_1, \ldots, I_k\}$$

   s.t.

$$\forall r \in \mathbb{R} \left(\exists p \in P_1(p(r) = 0) \implies r \in i(r)\right).$$

   This $\{I_1, \ldots, I_k\}$ is called an *isolating set of intervals* for the roots of P_1.
3. Because of their pairwise disjointness, $I_1, \ldots, I_k$ have a natural ordering determined, for instance, by comparing their lower-bound components. WLOG, assume

$$I_1 < I_2 < \ldots < I_k.$$

   This gives a "sketch" of a CAD of $\mathbb{R}^1$, with each interval I_i giving an approximation to a 0-dimensional cell (a *section*) which consists only of a root of a polynomial in P_1. If we were performing normal CAD without exploiting McCallum's Theorem, we would have to exactly represent these 0-dimensional cells, which may be irrational algebraic numbers, and construct stacks over them. The 1-dimensional cells of the P_1-induced CAD of $\mathbb{R}^1$ are those in between each adjacent pair of 0-dimensional cells, before the 0-dimensional cell contained in I_1 and after the 0-dimensional cell contained in I_k. As our isolating intervals are pairwise disjoint, they give us enough information, without any further refinement, to select sample points in the 1-dimensional cells. Thankfully, this means that we do not have to exactly represent any of the 0-dimensional cells; our approximations of them given by $I_1, \ldots, I_k$ are good enough.

4. As we are only interested in full-dimensional cells, we only need sample points *in between* adjacent I_i's, before I_1 and after I_k. Since every such region we will be sampling is an open subset of $\mathbb{R}^1$, we can choose these sample points all to be rational points.
5. In our example, we choose the following sample points to form S_1, our base collection of sample points of $\mathbb{R}^1$ with one point taken from each full-dimensional cell of a P_1-invariant CAD of $\mathbb{R}^1$:

$$S_1 = \{-1, 1\}.$$

Four Variants of Lifting With the projection and base phases completed, we turn to the lifting phase of (partial) CAD construction. To illustrate the use of our concrete AP-CAD stage, we will show how the following four distinct approaches to lifting differ when applied to deciding φ:

1. full-dimensional lifting without eliminating any full-dimensional cells,
2. full-dimensional lifting with standard partial CAD used to eliminate cells,
3. full-dimensional lifting with our AP-CAD theatre used to eliminate cells,
4. full-dimensional lifting with a combination of standard partial CAD and our AP-CAD theatre used to eliminate cells.

It will turn out that as we consider them in sequence, each subsequent lifting method will exhibit quite different behaviour in terms of the number of cells eliminated. In the end, full-dimensional lifting with a combination of standard partial CAD and our AP-CAD theatre will allow us to decide φ in the most efficient manner.

Lifting Variant I: All Full-dimensional Cells In this first variant of lifting, we will construct the entire full-dimensional CAD. In general, it is structured as a tree of sample points, each drawn from a full-dimensional cell. But, since we are deciding a purely $\exists$ **RCF** formula, we can ignore the tree structure and arrange our representation as a collection of sets of sample points, with one set of sample points for each CAD level. (There is still an implicit tree structure, however, as a sample point $\langle r_1, r_2, r_3 \rangle \in \mathbb{R}^3$ will be seen as a "child" of the sample point $\langle r_1, r_2 \rangle \in \mathbb{R}^2$, for instance.) Since our φ is 4-dimensional, and as we are only sampling rational points, we will end up with four sets of sample points, $S_1 \subset \mathbb{Q}^1$ (which we have already computed), $S_2 \subset \mathbb{Q}^2$, $S_3 \subset \mathbb{Q}^3$ and $S_4 \subset \mathbb{Q}^4$. At times we will only describe salient features of these additional sets of sample points, instead of presenting them explicitly, as they become large. We construct them as follows:

($\mathbb{R}^1 \mapsto \mathbb{R}^2$): To lift from $\mathbb{R}^1$ to $\mathbb{R}^2$, we iterate over our base set of sample points S_1 as follows (recall $|S_1| = 2$):
For each $q \in S_1$,

1. Form the univariate family $P_2[x_1 \mapsto q]$ by substituting[10] q for x_1 in P_2,
2. Compute a "sketch" of a CAD of $\mathbb{R}^1$ induced by $P_2[x_1 \mapsto q]$ (in the same manner we construced a "sketch" of a CAD of $\mathbb{R}^1$ induced by P_1 above through univariate real root isolation and isolating intervals), and select rational sample points x_i from each of its full-dimensional cells. For each sample point $x_i \in \mathbb{Q}$ selected, we then form a 2-dimensional sample point through extending q by x_i, obtaining $\langle q, x_i \rangle \in \mathbb{Q}^2$. Let $S_{2,q}$ be the set consisting of these sample points of the form $\langle q, x_i \rangle$. ($S_{2,q}$ then represents a full-dimensional stack over q.)

Finally, our set of 2-dimensional sample points S_2 is the union of these $S_{2,q}$ as follows:

$$S_2 = \bigcup_{q \in S_1} S_{2,q}.$$

Given P_2 and S_1 as computed above during the projection and base phases of deciding φ, $S_2 \subset \mathbb{Q}^2$ computed in this way will be s.t.

$$|S_2| = 14.$$

($\mathbb{R}^2 \mapsto \mathbb{R}^3$): We perform the next-dimensional analogue of the above procedure, this time working over the 14 sample points (each 2-dimensional) in S_2 and substituting them into P_3. After performing the relevant root isolation and sampling computations, this yields $S_3 \subset \mathbb{Q}^3$ s.t.

$$|S_3| = 40.$$

($\mathbb{R}^3 \mapsto \mathbb{R}^4$): Finally, we perform the next-dimensional analogue of the previous liftings, this time working over the 40 sample points (each 3-dimensional) in S_3 and substituting them into P_4. After performing the relevant root isolation and sampling computations, this yields $S_4 \subset \mathbb{Q}^4$ s.t.

$$|S_4| = 200.$$

So, lifting over every full-dimensional cell in our φ example ultimately results in having to compute 200 sample points in $\mathbb{Q}^4$, which will then be each substituted into $QF(\varphi)$ during the evaluation phase.

The coordinates of our sample points tend to become[11] more computationally unwieldy as we rise in dimension. For instance, here is one of these 200 sample points we computed for S_4.

$$\langle -1, -43/16, -119327/36200, 23133930249499/9896442880000 \rangle.$$

[10] Note that as q may in general be in $(\mathbb{Q} \setminus \mathbb{Z})$, we may have that $P_2[x_1 \mapsto q] \subset (\mathbb{Q}[x_2] \setminus \mathbb{Z}[x_2])$. This turns out to not cause any problems, and indeed can be avoided altogether without changing the real affine variety induced by $P_2[x_1 \mapsto q]$ by multiplying through the resulting univariate polynomials by the denominators of their rational coefficients.

[11] We have some ideas for methods enabling us to select sample points with smaller bit-width than those we often select from sectors now. Pursuing this remains as future work.

Here is one witness to φ contained in S_4, thus proving φ to be **true** over **RCF**:

$$\langle -1, 7/8, 501/410, 3917/410 \rangle.$$

Let us see how incorporating the methods of *partial* CAD during lifting can improve the situation by reducing the number of cells we must lift over.

Lifting Variant II: Classical Partiality In this variant of lifting, we will proceed in the manner of classical partial CAD (restricted to full-dimensional cells). This follows the basic algorithm described in **Section 3**. We recall the idea.

Beginning with our sample points $S_1 \subset \mathbb{Q}^1$ computed in the base phase, the "partiality" of this variant of lifting comes from the following process, which we follow for each $q \in S_1$: Before lifting over q, we will substitute q as a value for x_1 into $QF(\varphi)$ and examine the truth of the resulting formula $QF(\varphi)[x_1 \mapsto q]$. If $QF(\varphi)[x_1 \mapsto q]$ can by ground evaluation and propositional reasoning (cf. **Section 3**) be seen to be unsatisfiable, then we will eliminate q and avoid lifting over it. Dually, if $QF(\varphi)[x_1 \mapsto q]$ can be seen to be satisfiable by polynomial arithmetic and propositional reasoning, then we can stop the CAD process altogether and judge φ to be **true**. If we happen to eliminate *all* of our sample points, then we can judge φ to be **false**. This "partiality" is then performed in the analogous manner when lifting to each successive dimension.

As discussed in **Section 3**, classical partial CAD requires a sample point selection strategy. When partially lifting from $\mathbb{R}^i$ to $\mathbb{R}^{i+1}$, this specifies an enumeration of S_i, the sample points for the relevant cells of $\mathbb{R}^i$. We use a simple cell selection mechanism below, given by ordering the members of S_i by the lexicographic extension of the normal $<$ relation of $\mathbb{R}$ and then selecting the sample points from left to right.

($\mathbb{R}^1 \mapsto \mathbb{R}^2$): Performing the partial CAD method as described above results in no elimination of members of S_1.
We then substitute these two points as values for x_1 in P_2, and perform the root isolation and full-dimensional sample point selection computations. As before, this results in a total of 14 sample points in $\mathbb{Q}^2$. We use the following points:

$$\left\{ \begin{array}{r} \langle 1, -2 \rangle, \langle 1, -7/8 \rangle, \langle 1, -17/32 \rangle, \langle 1, -5/12 \rangle, \langle 1, -1/6 \rangle, \\ \langle 1, 3/4 \rangle, \langle 1, 43/16 \rangle, \langle -1, -43/16 \rangle, \langle -1, -3/4 \rangle, \langle -1, 1/6 \rangle, \\ \langle -1, 5/12 \rangle, \langle -1, 17/32 \rangle, \langle -1, 7/8 \rangle, \langle -1, 2 \rangle \end{array} \right\}.$$

($\mathbb{R}^2 \mapsto \mathbb{R}^3$): We perform the analogous partial lifting method for $\mathbb{R}^2 \mapsto \mathbb{R}^3$, this time working over our 14 sample points in $\mathbb{Q}^2$ computed above. In doing so, 7 out of the 14 sample points are eliminated. We are then left with only having to lift over the following set of 7 rational points in $\mathbb{Q}^2$:

$$\{\langle 1, 3/4 \rangle, \langle 1, 43/16 \rangle, \langle -1, 1/6 \rangle, \langle -1, 5/12 \rangle, \langle -1, 17/32 \rangle, \langle -1, 7/8 \rangle, \langle -1, 2 \rangle\}.$$

We then substitute these points, with each one giving values for x_1 and x_2, into P_3, and perform the root isolation and full-dimensional sample point

selection computations. This results in a total of only 20 sample points in $\mathbb{Q}^3$.

($\mathbb{R}^3 \mapsto \mathbb{R}^4$): We perform the analogous partial lifting method for $\mathbb{R}^3 \mapsto \mathbb{R}^4$, this time working over our 20 sample points in $\mathbb{Q}^3$ computed above. In doing so, 10 out of the 20 sample points are eliminated. We are then left with only having to lift over 10 rational points in $\mathbb{Q}^3$.

We then substitute these points, with each one giving values for x_1, x_2 and x_3, into P_4, and perform the root isolation and full-dimensional sample point selection computations. This results in a total of only 50 sample points in $\mathbb{Q}^4$.

Clearly, tremendous gains were made by employing partiality during lifting. Let us illustrate the differences between these first two lifting methods by comparing the cardinalities of the collections of sample points they retained at each dimension:

	Normal	Partial
$\mathbb{Q}^1$	2	2
$\mathbb{Q}^2$	14	7
$\mathbb{Q}^3$	40	10
$\mathbb{Q}^4$	200	50

Lifting Variant III: AP-CAD with Interval Theatre We now consider a lifting method which utilises our interval-based AP-CAD theatre defined in **Appendix B**. This will follow the "AP-CAD with Theatrical Lifting" algorithm (**Algorithm 1**) introduced within the main text of this paper, but instantiated upon our concrete theatre $\mathbb{T}$.

Recall that we defined $\mathbb{T}$ to be s.t.

$$\forall n \in \mathbb{N}^+ \ \ (\mathbb{T}(n) = \langle\langle \mathbb{S}, w\rangle, \mathbb{F}, \mathbb{P}\rangle) \, .$$

Thus, when performing AP-CAD lifting with $\mathbb{T}$, the same AP-CAD stage, $\langle\langle \mathbb{S}, w\rangle, \mathbb{F}, \mathbb{P}\rangle$ which we built to use some simple interval-based methods, will be applied at every dimension. The algorithm proceeds as follows:

$\mathbb{R}^1 \mapsto \mathbb{R}^2$: The covering width function w is applied to S_1 to yield $w(S_1) = 3$. This gives an upper-bound on the number of cell selections we will compute. As with the"AP-CAD with Theatrical Lifting" algorithm, we use j to represent the "step" in the cell selection processing. Initially, j is set to 1. Next, the cell selection function $\mathbb{S}$ is applied to S_1 with a step value of 1, yielding:

$$\mathbb{S}(S_1, 1) = S_1 = \{-1, 1\}.$$

So, the entire set of base sample points has been selected.

Next, the formula construction function $\mathbb{F}$ is executed upon φ and this selection of sample points. It yields the following formula:

$$\mathbb{F}(\varphi, \{-1, 1\}) = \left[\begin{array}{r} \exists x_1\ \exists x_2\ \exists x_3 \exists x_4 \\ (x_1x_4 + x_2x_4 + x_3x_2 < 0) \\ \wedge\ (x_2 > 0)\ \wedge\ (x_3 > 0)\ \wedge\ (x_4 > 0) \\ \wedge\ (x_3x_4 - x_4^2 + x_3^2 + 1 < 0) \\ \wedge\ (x_1 \geq -1)\ \wedge\ (x_1 \leq 1) \end{array}\right].$$

Finally, the $\exists$ **RCF** proof procedure $\mathbb{P}$ given by **RAHD**'s execution of the following proof strategy is executed upon $\mathbb{F}(\varphi, S_1)$:

```
[ simp-zrhs; run stable-simp; satur-lin;
  interval-cp(max-contractions := 30)].
```

This proof strategy is unable to reach a decision about $\mathbb{F}(\varphi, S_1)$ and returns **unknown**. This causes j to be incremented to 2, and the next step of the cell selection process is executed:

$$\mathbb{S}(S_1, 2) = \{-1\}.$$

Next, the formula construction function $\mathbb{F}$ is executed upon φ and this selection of sample points. It yields the following formula:

$$\mathbb{F}(\varphi, \{-1\}) = \left[\begin{array}{r} \exists x_1\ \exists x_2\ \exists x_3 \exists x_4 \\ (x_1x_4 + x_2x_4 + x_3x_2 < 0) \\ \wedge\ (x_2 > 0)\ \wedge\ (x_3 > 0)\ \wedge\ (x_4 > 0) \\ \wedge\ (x_3x_4 - x_4^2 + x_3^2 + 1 < 0) \\ \wedge\ (x_1 \geq -1)\ \wedge\ (x_1 \leq -1) \end{array}\right].$$

The $\exists$ **RCF** proof procedure $\mathbb{P}$ is executed upon $\mathbb{F}(\varphi, \{-1\})$, again returning **unknown**. This causes j to be incremented to 3, its current upper bound as determined by the covering width function w, and so a final cell selection will be executed upon S_1:

$$\mathbb{S}(S_1, 3) = \{1\}.$$

Next, the formula construction function $\mathbb{F}$ is executed upon φ and this selection of sample points. It yields the following formula:

$$\mathbb{F}(\varphi, \{1\}) = \left[\begin{array}{r} \exists x_1\ \exists x_2\ \exists x_3 \exists x_4 \\ (x_1x_4 + x_2x_4 + x_3x_2 < 0) \\ \wedge\ (x_2 > 0)\ \wedge\ (x_3 > 0)\ \wedge\ (x_4 > 0) \\ \wedge\ (x_3x_4 - x_4^2 + x_3^2 + 1 < 0) \\ \wedge\ (x_1 \geq 1)\ \wedge\ (x_1 \leq 1) \end{array}\right].$$

The $\exists$ **RCF** proof procedure $\mathbb{P}$ is executed upon $\mathbb{F}(\varphi, \{1\})$, and this time it is able to prove the constructed formula to be **false**. Thus, the sample point 1 can be eliminated from S_1 and we need not lift over it.

It is worth pausing and understanding why classical partial CAD was unable to eliminate 1 from S_1, yet this AP-CAD method succeeds. By inspecting the formula, we see that simple interval reasoning is enough to recognise the falsity of $\mathbb{F}(\varphi, \{1\})$, but classical partial CAD, performing only substitution, the evaluation of ground atoms, and propositional reasoning, does not recognise this.

Now, we isolate the relevant sample points induced by the univariate family $P_2[x_1 \mapsto -1]$ and continue onto the next dimension.

($\mathbb{R}^2 \mapsto \mathbb{R}^3 \mapsto \mathbb{R}^4$): It turns out that for the rest of the lifting process, our AP-CAD theatre is unable to eliminate any cells. This results in having to retain 20 sample points in $\mathbb{Q}^3$ and 100 sample points in $\mathbb{Q}^4$.

Thus, while this AP-CAD instance showed some promise in improving the efficiency of this example during ($\mathbb{R}^1 \mapsto \mathbb{R}^2$) lifting, in the end it resulted in having to lift over more cells than classical partial CAD did. Let us extend our previous table so that we may compare the cardinalities of the sets of retained sample points for the three variants of lifting seen thus far:

	Normal	Partial	Intvl. AP-CAD
$\mathbb{Q}^1$	2	2	1
$\mathbb{Q}^2$	14	7	7
$\mathbb{Q}^3$	40	10	20
$\mathbb{Q}^4$	200	50	100

But, notice the following: We did not employ at all the method of classical partial CAD during this AP-CAD lifting. That is, many of these cells the AP-CAD did not recognise to be eliminable may have been recognised to be eliminable by substitution, the evaluation of ground atoms and propositional reasoning. Let us see what happens when we combine these methods.

Lifting Variant IV: Classical Partial + AP-CAD In this final variant of lifting, we will first try to eliminate cells by the reasoning of classical partial CAD, and we will then apply our AP-CAD cell selection and elimination loop to the cells which survived.

($\mathbb{R}^1 \mapsto \mathbb{R}^2$): We begin with two sample points $S_1 = \{-1, 1\}$. Partial CAD elimination is unable to eliminate either of them. Our AP-CAD theatre is able to eliminate one of them, 1, as we saw before. So, we lift over -1 w.r.t. P_2 and compute 7 sample points in $\mathbb{Q}^2$

($\mathbb{R}^2 \mapsto \mathbb{R}^3$): We begin with the 7 sample points in $\mathbb{Q}^2$ and apply partial CAD elimination. This results in 2 sample points being eliminated. Our AP-CAD theatre is unable to eliminate any of them. We are then left with only having to lift over 5 points in $\mathbb{Q}^2$ w.r.t. $\mathbb{P}^3$. So, we lift over them and compute 14 sample points in $\mathbb{Q}^3$.

($\mathbb{R}^3 \mapsto \mathbb{R}^4$): Finally, we begin with the 14 sample points in $\mathbb{Q}^3$ and apply partial CAD elimination to them. This eliminates 7. We then lift over the remaining 7 points w.r.t. $\mathbb{P}_4$ and compute 35 sample points in $\mathbb{Q}^4$.

Thus, the combination of the cell elimination method of classical partial CAD coupled with our AP-CAD lifting led to the most efficient lifting variant, measured by the number of cells retained at each dimension, for this example. We may now complete our table comparing the cardinalities of the sets of sample points retained at each dimension:

	Normal	Partial	Intvl. AP-CAD	Partial + Intvl. AP-CAD
$\mathbb{Q}^1$	2	2	1	1
$\mathbb{Q}^2$	14	7	7	5
$\mathbb{Q}^3$	40	10	20	7
$\mathbb{Q}^4$	200	50	100	35

Experimental Conclusion In this experiment, we built a concrete instance of our Abstract Partial CAD framework making use of light-weight interval arithmetic reasoning and examined its efficacy. We compared in substantial detail four variants of lifting on a particular $\exists$ **RCF** formula φ. These four methods were:

1. full-dimensional lifting without eliminating any full-dimensional cells,
2. full-dimensional lifting with standard partial CAD used to eliminate cells,
3. full-dimensional lifting with our AP-CAD theatre used to eliminate cells,
4. full-dimensional lifting with a combination of standard partial CAD and our AP-CAD theatre used to eliminate cells.

As lifting is usually the most expensive aspect of a CAD-based decision method, we focused on a comparison between the number of cells one is forced to lift over by each of these variants.

For our example formula φ, we found the final method combining classical partial CAD and our AP-CAD instance to be the best, followed by classical partial CAD, then our AP-CAD instance working alone, and finally the method of normal full-dimensional CAD without any partiality. In all cases, the cost of the AP-CAD theatre/stage execution was miniscule, measuring no more than 0.01% of the total CPU time, as the cell selection, formula construction and interval reasoning employed were each of such a simple nature.

In general, we conclude that the final variant of lifting combining classical partial CAD and our AP-CAD instance at worst performs as well as partial CAD and at best performs substantially better. This conclusion is supported by experiments we have done with other example $\exists$ **RCF** formulas. Below we summarise in table form our findings on five examples (cf. [9]), the first being φ we worked through above:

		Normal	Partial	Intvl. AP-CAD	Partial + Intvl. AP-CAD
P1	$\mathbb{Q}^1$	2	2	1	1
	$\mathbb{Q}^2$	14	7	7	5
	$\mathbb{Q}^3$	40	10	20	7
	$\mathbb{Q}^4$	200	50	100	35
P2	$\mathbb{Q}^1$	16	8	0	0
	$\mathbb{Q}^2$	140	0	-	-
	$\mathbb{Q}^3$	664	-	-	-
P3	$\mathbb{Q}^1$	4	2	2	2
	$\mathbb{Q}^2$	20	5	10	5
	$\mathbb{Q}^3$	60	3	30	3
	$\mathbb{Q}^4$	120	6	60	6
P4	$\mathbb{Q}^1$	12	10	0	0
	$\mathbb{Q}^2$	88	19	-	-
	$\mathbb{Q}^3$	264	19	-	-
	$\mathbb{Q}^4$	1320	95	-	-
P5	$\mathbb{Q}^1$	8	3	4	3
	$\mathbb{Q}^2$	64	8	32	8
	$\mathbb{Q}^3$	512	8	56	8
	$\mathbb{Q}^4$	2560	40	1280	40

Finally, we wish to state two closing experimental observations.

First, the AP-CAD instance we constructed and experimented with in this section is but one of many (indeed, infinitely many) possible such instances. The fact that even such a simple[12] instance of the AP-CAD paradigm shows such promise is very encouraging.

Second, though we have been working solely with full-dimensional variants of CAD-based methods, AP-CAD may prove to be even more useful when it comes to full-on CAD-based methods which require irrational algebraic number computations. The reason is that through cell selection, formula construction and proof procedure execution, one has the ability to eliminate a set of *many* sample points all at once using AP-CAD, and in this way many irrational algebraic sample points may be eliminated with only *rational* number computations. To use our concrete AP-CAD instance as an example in the context of standard CAD not restricted to full-dimensional lifting, one has the potential to eliminate a set of sample points such as $\{-3, -\sqrt{2}, -1, \sqrt{2}, \sqrt{2}+3\sqrt{3}, 15\}$ simply by constructing and refuting a formula that only references this set of sample points using its minimal and maximal *rational* values, e.g., through a statement of the form $F \wedge (x_1 \geq -3 \ \wedge \ x_1 \leq 15)$ for some F. The ability to eliminate multiple irrational algebraic sample points simply through reasoning about formulas in-

[12] It is worth observing that the combination of classical partial CAD and our AP-CAD instance could actually be realised by a more intricate AP-CAD instance which performs the classical partial CAD reasoning itself. In this way, AP-CAD can be seen as a true generalisation of classical partial CAD.

volving rational numbers seems very promising for the extension of these ideas to unrestricted cell decompositions.