Reasoning Method between Polynomial Error Assertions

Peng Wu 1,2,*, Ning Xiong 2, Juxia Xiong 3 and Jinzhao Wu 1,3,*

1 School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China
2 School of Innovation, Design and Engineering, Mälardalen University, 72123 Västerås, Sweden; ning.xiong@mdh.se
3 School of Mathematics and Physics, Guangxi University for Nationalities, Nanning 530006, China; xiongjuxia@gxun.edu.cn
* Correspondence: 14112078@bjtu.edu.cn (P.W.); wujinzhao@gxu.edu.cn (J.W.)

Abstract: Error coefficients are ubiquitous in systems. In particular, errors in reasoning verification must be considered regarding safety-critical systems. We present a reasoning method that can be applied to systems described by the polynomial error assertion (PEA). The implication relationship between PEAs can be converted to an inclusion relationship between zero sets of PEAs; the PEAs are then transformed into first-order polynomial logic. Combined with the quantifier elimination method, based on cylindrical algebraic decomposition, the judgment of the inclusion relationship between zero sets of PEAs is transformed into judgment error parameters and specific error coefficient constraints, which can be obtained by the quantifier elimination method. The proposed reasoning method is validated by proving the related theorems. An example of intercepting target objects is provided, and the correctness of our method is tested through large-scale random cases. Compared with reasoning methods without error semantics, our reasoning method has the advantage of being able to deal with error parameters.

Keywords: formal method; reasoning method; system verification; polynomial; error control

1. Introduction

Formal verification includes two main approaches: model-checking and theorem-proving. Model-checking, proposed by Clarke and Emerson, verifies safety properties by searching the state space [1,2]. In contrast, theorem-proving uses logic and mathematical reasoning to verify the safety properties of a piece of software [3]. These two approaches have complementary advantages [4,5] and are widely used in industries [6–8]. Reasoning methods in theorem-proving belong to the field of deductive reasoning, comprising syntax, semantics, and reasoning rules; and numerous research studies have been conducted in this area. Labeled transition systems or their similar structures are widely used in the field of formal verification, which is a common technical method for describing the transition behavior of systems [9]. In particular, the states and transition conditions of systems are defined by two-valued logic propositions. However, two-valued or multi-valued logic is inadequate to describe the state of complex systems [10]. The real polynomial algebraic transition system extends the domain space of multivalued logic to the real number domain, which can more effectively describe complex systems and verify their safety properties [11]. Particularly in the field of hybrid system verification [12], numerous theories based on differential invariants have been proposed [13–15].

Semantics, grammar, and reasoning rules have been extensively studied in the field of reasoning methods [16]. However, the system parameters are sometimes not precisely determined in complex systems. In most cases, only the value ranges of these parameters are known [17,18], thereby making the reasoning and verification of these systems more complex. However, few reports on the semantic descriptions of errors have been published. For example, the Gröbner basis is widely used in polynomial-based theorem-proving methods. However, as the Gröbner basis of polynomial equations does not vary
continuously along with the parameters, it significantly increases the difficulty of dealing with error parameters in reasoning methods. In engineering, absolute accuracy is neither possible nor necessary. For example, when the temperature reaches 80 °C, the system stops heating; this is called a guard condition. However, precise measurements are not possible because the measurement often contains errors. Owing to the accumulation of multiple error variables, the system may undergo the wrong transitions. This is unacceptable for safety-critical systems. For example, a guard condition is \( f(x_1, x_2, \ldots, x_n) < 100 \), where \( x_1, x_2, \ldots, x_n \) are the values measured by instruments and \( f \) is a function of \( x_1, x_2, \ldots, x_n \). Because \( x_1, x_2, \ldots, x_n \) are all measured values, the calculations cannot be completely accurate. In this way, errors may be further accumulated in the calculation of the function \( f(x_1, x_2, \ldots, x_n) \). In other words, the calculation result may conclude that the guard condition \( f(x_1, x_2, \ldots, x_n) < 100 \) is satisfied, when, in fact, it is perhaps not correct. Especially in the case of the disturbance-sensitive function, minor parameter changes can lead to completely different results. However, several studies on reasoning methods involve error parameters. The authors previously proposed a reasoning method for linear error assertion (LEA) [19]. In this method, combined with the convex properties of LEA, the conclusion that vertexes of the precursor assertion are contained in the zero set of the successor assertion can be used to determine whether there is an implication relationship between precursor assertion and successor assertion. However, zero sets of nonlinear assertions may not be a convex set, indicating that the method reported in [19] is ineffective in dealing with nonlinear error assertion.

Error semantics based on interval numbers can be transformed into an inequality relationship. This implies that some theorems of inequality-proving may be promising for nonlinear error assertions. For example, \( \bar{a} = [a^-, a^+] \) can be transformed into: \( a^- \leq \bar{a} \leq a^+ \). Quantifier elimination methods used in reasoning methods between polynomial error assertions (PEAs) are introduced in this study. The inclusion relationships of the zero sets of the PEAs are transformed into the quantifier elimination problems of semi-algebraic first-order logic. Our reasoning method can be used to verify systems described by PEAs.

2. Preliminary

2.1. Error Intervals

Errors are common and inevitable. This section presents the error semantics represented by the intervals and their operation rules. Errors were quantitatively characterized using interval numbers. The interval numbers generalize real numbers and have been successfully applied in numerous areas [20, 21]. The definition and operation rules of the interval numbers are introduced below.

Let \( \bar{a} = [a^-, a^+] \), \( \bar{b} = [b^-, b^+] \), \( c \) be a constant.

For addition:

\[
\begin{align*}
\bar{c} + \bar{a} &= [a^- + c, a^+ + c] \\
\bar{a} + \bar{b} &= [a^- + b^-, a^+ + b^+] \tag{1}
\end{align*}
\]

For subtraction:

\[
\begin{align*}
\bar{a} - \bar{b} &= [a^- - b^+, a^+ - b^-] \\
\bar{a} - c &= [a^- - c, a^+ - c] \tag{2}
\end{align*}
\]

For multiplication:

\[
\bar{a} \bar{b} = [\min(a^- b^-, a^- b^+, a^+ b^-, a^+ b^+), \max(a^- b^-, a^- b^+, a^+ b^-, a^+ b^+)] \tag{3}
\]

In particular, when \( \bar{a} \geq 0 \), \( \bar{b} \geq 0 \), then:

\[
ab = [a^- b^-, a^+ b^+] \tag{4}
\]

when \( c > 0 \), then:

\[
c\bar{a} = [ca^-, ca^+] \tag{5}
\]
when $c < 0$, then:  
\[ c\mathbf{a} = [ca^+, ca^-] \tag{6} \]
and when $c = 0$, then:  
\[ c\mathbf{a} = [0, 0] = 0 \tag{7} \]

For division, when $c > 0$, then:  
\[ \frac{\mathbf{a}}{c} = \left[ \frac{a^-}{c}, \frac{a^+}{c} \right] \tag{8} \]
and when $c < 0$, then:  
\[ \frac{\mathbf{a}}{c} = \left[ \frac{a^+}{c}, \frac{a^-}{c} \right] \tag{9} \]

The error intervals generalize real numbers, where $a^- = a^+$, $\pi$ is the real number. However, the four arithmetic operations of the interval numbers defined above are irreversible. An example is provided as follows.

If $a$ is a real number, then the equation $a + b - b = a$ must hold. However, if $a$ and $b$ are interval numbers, the previous equation does not necessarily hold.

Let $\mathbf{a} = [a^-, a^+]$ and $\mathbf{b} = [b^-, b^+]$.

From the operation rules above, Formula (10) is obtained:

\[ \mathbf{a} + \mathbf{b} - \mathbf{b} = [a^+ + b^+, b^+ - b^-] - \mathbf{b} = [a^+ - b^+, a^+ + b^- - b^+] \neq \mathbf{a} \tag{10} \]

Formula (10) implies that the reasoning process is irreversible (that is, the conclusion of the reasoning is only one necessary condition, and not the necessary and sufficient condition for the premise) and does not imply that the four operation rules with interval numbers are incorrect in reasoning methods. The increase in the number of operations, with interval numbers in the reasoning process, may lead to meaningless results. Therefore, it is essential to try other methods in the reasoning process, and this study provides a possible approach.

2.2. Polynomial Error Assertion

In this section, we introduce some of the mathematical concepts that have been established and are involved in our method (Definitions 1, 2, and 3). We also introduce the definitions of PEA and the zero set of PEAs (Definitions 4 and 5) that are contributed by this study.

**Definition 1.** A real polynomial is a mathematical expression that comprises several terms. Each term includes one or more variables raised to power and is multiplied by a real coefficient. The general form of a polynomial with $m$ terms is given by Formula (11):

\[ f(x_1, x_2, \ldots, x_n) = \sum_{i=0}^{m} c_i x_1^{a_{i1}} x_2^{a_{i2}} \ldots x_n^{a_{in}} \tag{11} \]

Let $\mathbb{R}[x_1, \ldots, x_n]$ be the set of all real polynomials. An example of a real polynomial is as follows:

\[ f_1(x_1, x_2, x_3) = 3x_1^3 x_2 + x_1^2 + 2x_2 \tag{12} \]

**Definition 2 (Real zero set of real polynomials, i.e., zero set of polynomials).** Let $f(x_1, \ldots, x_n) \in \mathbb{R}[x_1, \ldots, x_n]$. The real zero set of $f(x_1, \ldots, x_2)$ is expressed as (13), denoted by $\text{Zero}(f)$:

\[ \text{Zero}(f) = \{(x_1, \ldots, x_n) | f(x_1, \ldots, x_2) = 0, \text{and} \ (x_1, \ldots, x_n) \in \mathbb{R}^n \} \tag{13} \]
Definition 3 (Real zero set of polynomials group). Let $\varphi$ be a polynomial group comprising $f_1, f_2, \ldots, f_n$, which is expressed as (14), and: $f_1 \in \mathbb{R}[x_1, \ldots, x_n], f_2 \in \mathbb{R}[x_1, \ldots, x_n], \ldots, f_n \in \mathbb{R}[x_1, \ldots, x_n]$.

$$
\varphi = \begin{cases} 
  f_1(x_1, \ldots, x_n) = 0 \\
  f_2(x_1, \ldots, x_n) = 0 \\
  \vdots \\
  f_n(x_1, \ldots, x_n) = 0 
\end{cases} \quad (14)
$$

The real zero set of $\varphi$, denoted by $\text{Zero}(\varphi)$, satisfies Formula (15).

$$
\text{Zero}(\varphi) = \text{Zero}(f_1) \cap \text{Zero}(f_2) \cap \ldots \cap \text{Zero}(f_n) \quad (15)
$$

The definition of the polynomial error assertion is provided below.

Definition 4 (Polynomial error assertion, i.e., PEA). $\overline{\varphi}$ is a PEA if $\overline{\varphi}$ and $f_1, f_2, \ldots, f_n$ satisfy Formulas (16) and (17), respectively.

$$
\overline{\varphi} = \begin{cases} 
  \overline{f}_1(x_1, \ldots, x_n) = 0 \\
  \overline{f}_2(x_1, \ldots, x_n) = 0 \\
  \vdots \\
  \overline{f}_n(x_1, \ldots, x_n) = 0 
\end{cases} \quad (16)
$$

$$
\overline{f}_i(x_1, \ldots, x_n) = \sum_{j=0}^{m} c_{ij}^{-} x_1^{a_{ij}^{-}} x_2^{a_{ij}^{-}} \ldots x_n^{a_{ij}^{-}}, i = 1, 2 \ldots n \\
\quad j = 0, \ldots, m \quad (17)
$$

Definition 5 (Real zero set of PEAs, zero set of PEAs). $\overline{\varphi}$ is a PEA. The real zero set of $\overline{\varphi}$, denoted by $\text{zero}(\overline{\varphi})$, satisfies Formula (18).

$$
\text{zero}(\overline{\varphi}) = \text{zero}(\overline{f}_1) \cap \text{zero}(\overline{f}_2) \cap \ldots \cap \text{zero}(\overline{f}_n) \quad (18)
$$

In addition, $\text{zero}(\overline{f}_1), \text{zero}(\overline{f}_2), \ldots, \text{zero}(\overline{f}_n)$ satisfies Formula (19).

$$
\text{zero}(\overline{f}_i) = \left\{ (x_1, \ldots, x_n) \mid \forall c_{ij} \in c_{ij}, \text{ and } \sum_{j=0}^{m} c_{ij} x_1^{a_{ij}} x_2^{a_{ij}} \ldots x_n^{a_{ij}} = 0, \text{ and } (x_1, \ldots, x_n) \in \mathbb{R}^n \right\} \quad (19)
$$

From Definition 5, $\text{zero}(\overline{\varphi})$ contains all possible solutions of Formula (19) owing to the error coefficients. Therefore, all possible states of the system generated by the error parameters can be defined using $\text{zero}(\overline{\varphi})$. In engineering, if some system states are not found to be under $\text{zero}(\overline{\varphi})$, this indicates that there are non-error factors affecting the system. Moreover, determining whether there are some mechanical failures in the system is necessary.

2.3. Implication Relationships

The implication relationship between assertions is the fundamental rule in the reasoning method based on theorem-proving. We introduce the equivalent expression of the implication relationship between polynomials and zero sets in Theorems 1 and 2, respectively.

Theorem 1. $\varphi_1$ implies $\varphi_2$ (denoted as $\varphi_1 \models \varphi_2$) iff $\text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2)$, where $\varphi_1$ and $\varphi_2$ are the two polynomials.

In Theorem 1, the comprehensible statement of the implication relationship ($\varphi_1 \models \varphi_2$) is as follows: if $\varphi_1$ is true, then $\varphi_2$ must be true. For example, $\varphi_1 : x - 1 = 0$, $\varphi_2 :$
\[ x^2 - 3x + 2 = 0, \text{ then } \varphi_1 \models \varphi_2 \text{ must be true. Apparently, if the formula } x - 1 = 0 \text{ holds, then } x^2 - 1 = 0 \text{ must be true. The formula } \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \text{ is obtained using Theorem 1. Indeed, } \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \text{ must hold because } \{1\} \subseteq \{1, 2\}. \]

**Theorem 2.** \( \varphi_1 \) implies \( \varphi_2 \) (denoted as \( \varphi_1 \models \varphi_2 \)) iff \( \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \), where \( \varphi_1 \) and \( \varphi_2 \) are the two PEA.

The significance of \( \text{Zero}(\varphi) \) lies in all possible system states produced by error coefficients. Conversely, if the security state zone of the system is defined by \( \text{Zero}(\varphi_2) \), and, at a certain moment, the system state \( x = (x_1, x_2, \ldots, x_n) \) satisfies \( (x_1, x_2, \ldots, x_n) \notin \text{Zero}(\varphi_2) \), then state \( x \) is unsafe for the system. If the security state zone of the system is defined by \( \text{Zero}(\varphi_2) \), and if \( \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \) is held, it implies that all states in \( \text{Zero}(\varphi_1) \) are safe.

### 3. Problem Descriptions

Let \( \varphi_1 \) and \( \varphi_2 \) be two PEA; we must determine whether \( \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \) holds. This question has safety significance in safety-critical systems. For example, considering the safety states of the system are designed using \( \text{Zero}(\varphi_2) \) according to the given error coefficients, the mechanical or electrical capability of the system meets the given design requirements and error tolerances. However, this does not mean that all possible error assignments can be met for the system design. For example, when extreme weather conditions occur, one of the error coefficients exceeds the given error range. At this time, it is necessary to identify whether the system can still not exceed the design requirements under the new parameter assignments. When \( \varphi_1 \) and \( \varphi_2 \) are both non-linear PEA, the vertex judgment method (VJM) in [19] is invalid. The idea of VJM is that \( \varphi_1 \models \varphi_2 \) is true if all vertices \( (v_1, v_2, \ldots, v_n) \) \( \varphi_1 \) satisfy the formula \( v_1 \in \text{Zero}(\varphi_2), v_2 \in \text{Zero}(\varphi_2), \ldots, v_n \in \text{Zero}(\varphi_2) \). However, VJM is based on convex conditions \( \text{Zero}(\varphi_2) \). The convex set has a property that all points inside it (including its boundary) can be generated finitely by its vertices. Moreover, if all the vertices of \( \text{Zero}(\varphi_1) \) are within \( \text{Zero}(\varphi_2) \) (including the boundary), then \( \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \) must hold. Unfortunately, \( \text{Zero}(\varphi_2) \) is not a convex set when \( \varphi_2 \) is a nonlinear PEA. For example, in Formula (45) in Section 5, \( \text{Zero}(\varphi_2) \) is not a convex set, and \( \text{Zero}(\varphi_2) \) is shown as the red part in Figure 1. Obviously, both A and B are inside \( \text{Zero}(\varphi_2) \), but some points on the line segment of AB are no longer inside \( \text{Zero}(\varphi_2) \).

![Figure 1. An example of a non-convex set.](image)

Therefore, VJM is invalid for PEA. For example, Figure 2a clearly shows that the rectangle ABCD is a convex set, and points A, B, C, and D are the vertices of the rectangle. When vertices A, B, C, and D are all inside the diamond-shaped EFGH, it can be concluded that all the points inside the rectangle ABCD are inside the diamond-shaped EFGH. However, the closed area formed by the line segments EF, FG, GH, HE (the area is shorted for EFGH) in Figure 2b is also a convex set, as seen in the red area in Figure 1.
Although vertices A, B, C, and D of the rectangle ABCD are all inside the non-convex EFGH (including the boundary), the red part in the rectangle ABCD remains outside EFGH.

![Figure 2](image-url)

**Figure 2.** An example of a failure for VJM.

The following section introduces a method of identifying the relationship between non-linear PEAs and the mathematical definitions involved.

4. Reasoning Method between PEAs

4.1. Two Mathematical Definitions

The constant coefficients—the semi-algebraic system (CSS) [22] and first-order polynomial logic formulas [14]—are introduced in Definitions 6 and 7, respectively. After Definition 6, the method of transforming PEA into a CSS is provided.

**Definition 6 (CSS).** Formula (20) represents the general form of a CSS:

\[
\begin{align*}
& p_1(x_1, \ldots, x_n) = 0, \ldots, p_s(x_1, \ldots, x_n) = 0, \\
& g_1(x_1, \ldots, x_n) \geq 0, \ldots, g_l(x_1, \ldots, x_n) \geq 0, \\
& g_{l+1}(x_1, \ldots, x_n) > 0, \ldots, g_{m}(x_1, \ldots, x_n) > 0, \\
& h_{1}(x_1, \ldots, x_n) \neq 0, \ldots, h_{m}(x_1, \ldots, x_n) \neq 0,
\end{align*}
\]

Formula (20) represents the general form of a CSS:

\[
\begin{align*}
& p_1(x_1, \ldots, x_n) = 0, \ldots, p_s(x_1, \ldots, x_n) = 0, \\
& g_1(x_1, \ldots, x_n) \geq 0, \ldots, g_l(x_1, \ldots, x_n) \geq 0, \\
& g_{l+1}(x_1, \ldots, x_n) > 0, \ldots, g_{m}(x_1, \ldots, x_n) > 0, \\
& h_{1}(x_1, \ldots, x_n) \neq 0, \ldots, h_{m}(x_1, \ldots, x_n) \neq 0,
\end{align*}
\]

Among them, \( p_1, \ldots, p_s, g_1, \ldots, g_l, h_1, \ldots, h_m \) are all real coefficient polynomials. CSS has a wide range of applications in terms of geometry, topology, and differential dynamics.

The given PEA can be transformed into a CSS. The general conversion method is as follows.

A PEA comprises multiple error polynomials, \( \tilde{f}(x_1, \ldots, x_n) \) is the error polynomial in the general form, as in Formula (17). The equivalent CSS is then shown in Formula (21):

\[
\begin{align*}
& \sum_{j=0 \ldots m} c_j x_1^{a_{1j}} x_2^{a_{2j}} \ldots x_n^{a_{nj}} = 0, \\
& c_1^+ \geq c_1, \ldots, c_j^+ \geq c_j, \\
& c_1 \geq c_1^- \ldots, c_j \geq c_j^-.
\end{align*}
\]

Thus, a PEA can be transformed into an equivalent CSS.

**Definition 7.** A first-order polynomial logic comprises the following four parts:

1. Terms of polynomial, which comprise these variables \((x_1, \ldots, x_n) or x, y, z \ldots \) in \(\mathbb{R}\);
2. Relational symbols: \(=, \neq, <, \leq, >, \geq \ldots \);
3. Boolean connectives: \(\wedge, \vee, \neg, \rightarrow, \leftrightarrow \);
4. Quantifiers: \(\forall, \exists\).

For example, Formula (22) is a first-order polynomial logic:

\[
\forall x \forall y((x^2 + y^2 < 4) \land (x^2 + y^2 > 1))
\]
Accordingly, a CSS can be equivalently transformed into a first-order polynomial logic, as shown in Formula (23):

\[
\begin{aligned}
x^2 + y^2 &< 4 \\
x^2 + y^2 &> 1 \\
x &> 0
\end{aligned} \iff (x^2 + y^2 < 4) \land (x^2 + y^2 > 1) \land (x > 0)
\tag{23}
\]

### 4.2. Quantifier Elimination

This section briefly introduces the quantifier elimination method, based on cylindrical algebraic decomposition (CAD). Tarski modified the reasoning process based on first-order logic to judge the inclusion relationship of specific sets. However, Tarski’s algorithm has a high time complexity for the problem of quantifier elimination [23]. The CAD algorithm was first proposed by Collins [24] and has been further improved by many researchers. At present, it is widely used in quantifier elimination problems. Xia et al. first implemented the CAD algorithm on Maple [25]. The CAD algorithm can equivalently convert first-order polynomial logic to judge the inclusion relationship of specific sets. However, Tarski's algorithm has high time complexity for the problem of quantifier elimination [23].

The CAD algorithm applies the projection operator Proj, thereby obtaining several first-order logic formulas. Hence, an arbitrary point in each cell can be selected to verify each subformula in this first-order polynomial logic. The algorithm first selects an appropriate projection operator (Proj), which projects a first-order logic formula with \( r \) variables to a first-order logic formula with only \( r - 1 \) variables. Then, it iteratively applies the projection operator Proj, thereby obtaining several first-order logic formulas with \( r - 1, r - 2, \ldots, 1 \) variables. Subsequently, the real root isolation algorithm is applied to these first-order logic formulas, such that the \( \mathbb{R}^r \) space is divided into several cells. All the points in a certain cell have the same sign on each subformula in this first-order polynomial logic. Hence, an arbitrary point in each cell can be selected to verify each formula in first-order polynomial logic, thereby providing an equivalent quantifier-free formula. The quantifier elimination method has been applied to many contexts [26–28]. Certain scholars have made some progress in the poly-complexity quantifier elimination method in the field of symbolic computation [29]. In this study, \( \text{QE}(\ ) \) denotes the quantifier elimination method.

For example, \( f \) is a first-order polynomial logic, such as in Formula (24). After eliminating the quantifier for \( f \) (denoted as \( \text{QE}(f) \)), the equivalent non-quantifier formula of \( f \) is as Formula (25):

\[
f = \exists y((x^2 + y^2 < 1) \land (-2 < x^2 + y < 2))
\tag{24}
\]

\[
f \rightarrow \text{QE}(f) \iff -1 < x < 1
\tag{25}
\]

### 4.3. Reasoning Method between PEAs

This section introduces some theorems proposed in this study (Lemma 1, Theorem 3, and Definitions 8 and 9), and the steps of the reasoning method between PEAs.

**Lemma 1.** \( \overline{f}_1 \) and \( \overline{f}_2 \) are two error polynomials, expressed in Formula (26), and are arranged in the same alphabetical order. If \( \overline{c}_1 \subseteq \overline{c}_2 \), \( i = 0, 1, \ldots, m \) and \( \alpha_{11} = \alpha_{21}, \alpha_{12} = \alpha_{22}, \ldots, \alpha_{1m} = \alpha_{2m} \) are true, then \( \text{Zero}(\overline{f}_1) \subseteq \text{Zero}(\overline{f}_2) \) must hold.

\[
\begin{aligned}
\overline{f}_1(x_1, \ldots, x_n) &= \sum_{j=0}^{m} \overline{c}_{1j} x_1^{\alpha_{1j1}} x_2^{\alpha_{1j2}} \cdots x_n^{\alpha_{1jm}} \\
\overline{f}_2(x_1, \ldots, x_n) &= \sum_{j=0}^{m} \overline{c}_{2j} x_1^{\alpha_{2j1}} x_2^{\alpha_{2j2}} \cdots x_n^{\alpha_{2jm}}
\end{aligned}
\tag{26}
\]

**Proof.** Let \( \overline{c}_{1j} = [c_{1j}, c_{1j}^+] \), \( \overline{c}_{2j} = [c_{2j}, c_{2j}^+] \). From \( \overline{c}_{1j} \subseteq \overline{c}_{2j} \), the Formula (27) is obtained:

\[
\overline{c}_{2j} = \overline{c}_{1j} \cup [c_{2j}, c_{2j}^+] \cup [c_{1j}, c_{1j}^+]
\tag{27}
\]

According to Definition 5, Formula (28) can be obtained:
Definition 8. Let $\varphi$ be the CSS equivalent to $\overline{\varphi}$, where $\overline{\varphi}$ is a PEA.

For example, $\overline{\varphi} : y^2 - [c_1^-, c_1^+] x^2 + x + [c_1^-, c_1^+] = 0$. The equivalent CSS of $\varphi$ is represented by Formula (30):

$$\varphi = \begin{cases} y^2 - c_1 x^2 + x + c_2 = 0 \\ c_1^- \leq c_1 \leq c_1^+ \\ c_2^- \leq c_2 \leq c_2^+ \end{cases}$$

(30)

Definition 9. $\varphi_\Delta$ is a real polynomial equation system that replaces all error coefficients in $\overline{\varphi}$ with symbols.

For example, $\overline{\varphi} : y^2 - [c_1^-, c_1^+] x^2 + x + [c_1^-, c_1^+] = 0$. Formula (31) demonstrates the symbolized Formula $\varphi_\Delta$:

$$\varphi_\Delta : y^2 - c_1 x^2 + x + c_2 = 0$$

(31)

Theorem 4. $\overline{\varphi_1}$ and $\overline{\varphi_2}$ are two PEs, and the error coefficients in $\overline{\varphi_1}$ and $\overline{\varphi_2}$ are provided. The equivalent CSS corresponding to $\overline{\varphi_2}$ is $\varphi_2$, presented in Definition 8. The symbolized formula of $\overline{\varphi_1}$ is $\varphi_{1\Delta}$, as expressed in Definition 9.

Assuming that all the error coefficients of $\overline{\varphi_1}$ are $c_{11} = [c_{11}^-, c_{11}^+]$, $c_{12} = [c_{12}^-, c_{12}^+]$, ..., $c_{1n} = [c_{1n}^-, c_{1n}^+]$ and all error coefficients of $\overline{\varphi_2}$ are $c_{21} = [c_{21}^-, c_{21}^+]$, $c_{22} = [c_{22}^-, c_{22}^+]$, ..., $c_{2m} = [c_{2m}^-, c_{2m}^+]$.

Then, if Formula (32) holds, $Zero(\overline{\varphi_1}) \subseteq Zero(\overline{\varphi_2})$ must be true:

$$\left\{ c_{11}, \ldots, c_{1n} \left| (c_{11}^--c_{11}^- \leq c_{11} \leq c_{11}^+) \wedge \ldots \wedge (c_{1n}^--c_{1n}^- \leq c_{1n} \leq c_{1n}^+) \right\} \subseteq \left\{ c_{11}, \ldots, c_{1n}, Q\varepsilon(x_1, \ldots, x_n, \exists x_1 \in [c_{21}^-, c_{21}^+], \ldots, \exists x_m \in [c_{2m}^-, c_{2m}^+], \overline{\varphi_1} = \overline{\varphi_2} \right\}$$

(32)

Proof. Let $\overline{\varphi_{1\Delta}} = (\varphi_{1\Delta} \wedge \forall c_{11}, \ldots, c_{1n} : (c_{11}, \ldots, c_{1n} \in [c_{11}^-, c_{11}^+] \wedge \ldots \wedge (c_{1n}^--c_{1n}^- \leq c_{1n} \leq c_{1n}^+)) \subseteq \left\{ c_{11}, \ldots, c_{1n}, Q\varepsilon(x_1, \ldots, x_n, \exists x_1 \in [c_{21}^-, c_{21}^+], \ldots, \exists x_m \in [c_{2m}^-, c_{2m}^+], \overline{\varphi_{1\Delta}} = \overline{\varphi_2} \right\}.$

Because the quantifier elimination method ($QE(\cdot)$) based on CAD can generate an equivalent quantifier-free logic constraint, Formula (33) can be obtained:

$$\overline{\varphi_{1\Delta}} = \overline{\varphi_2}$$

(33)

According to Theorem 2, Formula (34) holds:

$$Zero(\overline{\varphi_{1\Delta}}) \subseteq Zero(\overline{\varphi_2})$$

(34)
Conversely, according to Lemma 1 and Formula (32), we obtain Formula (35):

$$\text{Zero}(\varphi_1) = \text{Zero}(\varphi_{1s} \land (c_{11} \leq c_{11} \leq c_{11}^+) \land \ldots \land (c_{in} \leq c_{in} \leq c_{in}^+)) \subseteq \text{Zero}(\varphi_{2s})$$

(35)

From Formulas (34) and (35), Zero(\varphi_1) \subseteq Zero(\varphi_2) holds.

Based on Theorem 3, we can obtain the reasoning method for PEAs. Steps 1–4 can determine whether Zero(\varphi_1) \subseteq Zero(\varphi_2) is true. Some of the symbols used in the following steps are represented as follows:

\(\varphi_1\) and \(\varphi_2\) are two PEAs. All the error coefficients of \(\varphi_1\) and \(\varphi_2\) are \(c_{11} = [c_{11}, c_{11}^+], c_{12} = [c_{12}, c_{12}^+], \ldots, c_{in} = [c_{in}, c_{in}^+], c_{21} = [c_{21}, c_{21}^+], c_{22} = [c_{22}, c_{22}^+], \ldots, c_{2m} = [c_{2m}, c_{2m}^+]\), respectively. The steps to verify whether Zero(\varphi_1) \subseteq Zero(\varphi_2) holds are as follows:

1. Convert \(\varphi_1, \varphi_2\) into the symbolized formula \(\varphi_{1s}, \varphi_{2s}\) according to Definition 9.
2. Combine \(\varphi_{1s}, \varphi_{2s}\), and the symbol coefficients of \(\varphi_2\) to form a first-order polynomial logic, as expressed in Formula (36):

$$\varphi_{1s} \land \varphi_{2s} \land (c_{21} \leq c_{21} \leq c_{21}^+) \land \ldots \land (c_{2m} \leq c_{2m} \leq c_{2m}^+)$$

(36)

3. Convert \((c_{21} \leq c_{21} \leq c_{21}^+) \land \ldots \land (c_{2m} \leq c_{2m} \leq c_{2m}^+)\) from Formula (36) into the representation with the existential quantifier \(\exists\). Consequently, eliminate the quantifier for the formula \(\text{QE}(\forall x_1, \ldots, \forall x_n, \exists c_{21} \in [c_{21}, c_{21}^+], \ldots, \exists c_{2m} \in [c_{2m}, c_{2m}^+], (\varphi_{1s} = \varphi_{2s})\) to obtain a first-order constraint containing only \(c_{11}, \ldots, c_{in}\). The result is represented by Formula (37):

$$f_1(c_{11}, \ldots, c_{in}) \lor \ldots \lor f_L(c_{11}, \ldots, c_{in})$$

(37)

4. Finally, we determine whether Formula (38) holds. If (38) holds, then Zero(\varphi_1) \subseteq Zero(\varphi_2) holds:

$$\{c_{11}, \ldots, c_{in} | (c_{11} \leq c_{11} \leq c_{11}^+) \land \ldots \land (c_{in} \leq c_{in} \leq c_{in}^+)\} \subseteq \{c_{11}, \ldots, c_{in} | f_1(c_{11}, \ldots, c_{in}) \lor \ldots \lor f_L(c_{11}, \ldots, c_{in})\}$$

(38)

When there are multiple variables in Formula (37), the interval operation rules introduced in Section 2 may be needed to judge Formula (38).

A simple example is presented below to introduce the reasoning method between PEAs.

### 4.4. Simple Example

\(\varphi_1\) and \(\varphi_2\) are two PEAs as Formula (39). Now, we must judge whether Zero(\varphi_1) \subseteq Zero(\varphi_2) holds:

$$\begin{align*}
\varphi_1 &= \begin{cases}
    x^2 + y^2 \leq [1, 4] & x \in [0, \infty) \\
    y \leq [0, \infty)
  \end{cases} \\
\varphi_2 &= \begin{cases}
    (x - 1)^2 + y^2 \leq [0, 6] & x \in [0, \infty) \\
    y \leq [0, \infty)
  \end{cases}
\end{align*}$$

(39)

Based on the steps described in Section 4.3, the specific steps for the example are as follows:

1. The symbolized \(\varphi_1\) and \(\varphi_2\) are shown in Formula (40):

$$\begin{align*}
\varphi_{1s} &= \begin{cases}
    x^2 + y^2 = a & x \geq 0 \\
    y \geq 0
  \end{cases} \\
\varphi_{2s} &= \begin{cases}
    (x - 1)^2 + y^2 = b & x \geq 0 \\
    y \geq 0
  \end{cases}
\end{align*}$$

(40)

2. According to Step 2 in Section 4.3, Formula (41) can be obtained:

$$\varphi_{1s} \land \varphi_{2s} \land (c_{21} \leq c_{21} \leq c_{21}^+) \land \ldots \land (c_{2m} \leq c_{2m} \leq c_{2m}^+) = (x^2 + y^2 = a) \land (x \geq 0) \land (y \geq 0) \land (x - 1)^2 + y^2 = b) \land 0 \leq b \leq 6$$

(41)

3. Using the call quantifier elimination algorithm, Formula (42) can be obtained:
\[
Q\mathbb{E}(\forall x_1, \ldots, \forall x_n, \exists c_{21} \in [c_{21}^1, c_{21}^2], \ldots, \exists c_{2m} \in [c_{2m}^1, c_{2m}^2], \varphi_{1x}) = \varphi_{2x} = \\
Q\mathbb{E}(\forall x_1, \ldots, \forall x_n, \exists b \in [0, 6], ((x^2 + y^2 = a) \land (x \geq 0) \land (y \geq 0)) \big| = (((x - 1)^2 + y^2 = b) \land (x \geq 0) \land (y \geq 0)) = (42) \]

4. Finally, \([1, 4] \subseteq a \leq 5\), i.e., \(1 \leq a \leq 4 \subseteq a \leq 5\) is found to be true. Therefore, \(\text{Zero}(\mathcal{P}_1) \subseteq \text{Zero}(\mathcal{P}_2)\) is true.

The Maple2020 codes and the result of Step 3 are as follows:

```maple
R := PolynomialRing([x, y, a, b]);
with (RegularChains);
with (SemiAlgebraicSetTools);

f := &\& A [x, y], &E [b], (((y^2 + x^2 = a) \&and (0 <= x)) \&and (0 <= y)) \&implies (((x - 1)^2 + y^2 <= b) \&and (0 <= b)) \&and (b <= 6));
out := QuantifierElimination(f, 'output' = 'rootof');
out := a <= 5
```

Figures 3–6, generated from Python codes, show the conclusion more clearly. The red part of Figure 3 is the area of \(\text{Zero}(\mathcal{P}_1)\). The blue part of Figure 4 is the area of \(\text{Zero}(\mathcal{P}_2)\).

![Figure 3. The area of Zero(\(\mathcal{P}_1\)).](image-url)

![Figure 4. The area of Zero(\(\mathcal{P}_2\)).](image-url)

![Figure 5. The mixture of Figures 3 and 4.](image-url)
Figure 5. The mixture of Figures 3 and 4.

Figure 6. The situation regarding \( a = 5 \).

Does \( \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \) hold? We can mix Figures 3 and 4 to observe. Figure 5 is obtained by mixing Figures 3 and 4. As shown in Figure 5, the red part (\( \text{Zero}(\varphi_1) \)) is all inside \( \text{Zero}(\varphi_2) \). Furthermore, we are also interested in the boundary situation when \( a = 5 \) (variable “a” comes from Formula (40)). According to the calculation result from the Maple codes above in Section 4.4, we can conclude that when \( a <= 5 \), \( \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \) holds. Figure 6 shows the situation when \( a = 5 \).

In Figure 6, \( \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \) barely holds with the boundary condition \( (a = 5) \). This shows that, in this case, the results \( (a <= 5) \) obtained by our method are complete.

5. Example of Target Interception

5.1. Interception Verification

This section provides an example of the reasoning method between PEAs in the industry and the codes in Maple.

A self-powered flying target descends from a height of 10 km. Assuming that the expected landing point is the origin of the coordinates, the target flight path is always on a plane. The target starts to land at a horizontal distance \( s \) of 80 km from the point of origin, and a height \( h \) of 10 km in the vertical direction. After radar monitoring and system simulation, the landing trajectory of the target conforms to a PEA, which is represented by \( \varphi_1 \) in Formula (43):

\[
\varphi_1 = \left\{ y = \frac{10h}{s^2}x^3 - \frac{15h}{s^4}x^4 + \pi \cdot \frac{6h}{s^5}x^5 \right\} \quad (43)
\]

From Formula (43), \( \text{Zero}(\varphi_1) \) is obtained as shown in Formula (44):

\[
\text{Zero}(\varphi_1) = \left\{ (x,y) \left| \forall a \in [0.99,1.01], \frac{10h}{s^2}x^3 - \frac{15h}{s^4}x^4 + \pi \cdot \frac{6h}{s^5}x^5 - y = 0, 45 \leq x \leq 80 \right. \right\} \quad (44)
\]

At this point, it is necessary to attempt to intercept the target at 45 km from the point of origin in the horizontal direction. This distance (45 km from the origin) is to ensure safety. This is because being too close to the ground or point of origin may make the interception process meaningless, even if it is successfully intercepted. If the target is a missile with certain lethality, enough safe distance is necessary. Therefore, a group of intercepting objects is launched from the ground, which is designed to satisfy \( \varphi_2 \) (a PEA with three degrees), as shown in Formula (45):

\[
\varphi_2 = \left\{ y = \frac{3h}{s^2}x^2 - \frac{2h}{s^3}x^3 + 0.3 \right\} \quad (45)
\]

From Formula (45), \( \text{Zero}(\varphi_2) \) can be obtained as Formula (46):

\[
\text{Zero}(\varphi_2) = \left\{ (x,y) \left| \forall b \in [0.95,1.05], \frac{3h}{s^2}x^2 - \frac{2h}{s^3}x^3 + 0.3 - y = 0, 45 \leq x \leq 80 \right. \right\} \quad (46)
\]
Thereafter, we must determine whether the flying object can be intercepted under sufficiently dense launching of the interceptors. When the set of trajectories of the group of interceptors includes all possible trajectories of the target, the target can be intercepted. That is, \( \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \) holds.

The specific steps and their results are provided below, according to the method discussed in Section 4.3.

1. Convert \( \varphi_1, \varphi_2 \) in Formulas (43) and (45), respectively, into the symbolized \( \varphi_{1s}, \varphi_{2s} \) as shown in Formula (47).

\[
\varphi_{1s} = \left\{ \begin{array}{ll}
y = \frac{10^6}{45} x^3 \frac{15^5}{45} x^4 + a \cdot \frac{66}{45} x^5 & 45 \leq x \leq 80 \\
\varphi_{2s} = \left\{ y = \frac{36}{45} x^2 - b \cdot \frac{28}{45} x^3 + 0.3 \right. & 45 \leq x \leq 80
\end{array} \right.
\tag{47}
\]

2. By combining \( \varphi_{1s}, \varphi_{2s} \), and the error coefficients of \( \varphi_2 \), Formula (48) can be obtained.

\[
\varphi_{1s} \land \varphi_{2s} \land (c_{21} \leq c_{21} \leq c_{2n}) \land \ldots \land (c_{2m} \leq c_{2m} \leq c_{2m}) = \\
(y = \frac{10^6}{45} x^3 - \frac{10^6}{45} x^4 + a \cdot \frac{66}{45} x^5) \land (45 \leq x \leq 80) \land (y = \frac{36}{45} x^2 - b \cdot \frac{28}{45} x^3 + 0.3) \land (0.95 \leq b \leq 1.05)
\tag{48}
\]

3. The universal quantifier \( \forall \) acts on \( x \) and \( y \), and the existence quantifier \( \exists \) acts on \( b \) in Formula (48). Subsequently, we call the quantifier elimination algorithm to obtain Formula (49):

\[
\text{QE}(\forall x_1, \ldots, \forall x_n, \exists c_21 \in [c_{21}, c_{21}'], \ldots, \exists c_{2n} \in [c_{2n}, c_{2n}'], \varphi_{1s}) = \varphi_{2s}) = \\
\text{QE}(\forall (x, y), \exists b \in [0.95, 1.05], (y = \frac{10^6}{45} x^3 - \frac{15^5}{45} x^4 + a \cdot \frac{66}{45} x^5) \land (45 \leq x \leq 80) \land (y = \frac{36}{45} x^2 - b \cdot \frac{28}{45} x^3 + 0.3)) = 0.8983 \leq a \leq 1.0129
\tag{49}
\]

4. Because \( \exists \varphi = [0.99, 1.01] \subseteq [0.9883, 1.01129] \), we can conclude that \( \text{Zero}(\varphi_1) \subseteq \text{Zero}(\varphi_2) \) is true. That is, \( \varphi_1 \mid= \varphi_2 \) holds.

In Step 3, the quantifier elimination algorithm (QE) is available in Maple2020. However, it can only handle first-order polynomial logic with integer coefficients. Hence, Formula (49) must be multiplied by an appropriate integer before calling the quantifier elimination method to obtain Formula (49) without floating-point numbers. Subsequently, the conclusion of Step 4 must divide this integer (the integer is 6000 in this case), which is not repeated here. The Maple codes of Step 3 are as follows:

with (RegularChains);
with (SemiAlgebraicSetTools);
R := PolynomialRing ([x, y, a, b]):
f := &A [x, y, &E [b], ' &amp; &and (a*x^5 - 1,200,000*x^4 + 64,000,000*x^3 - 327,680,000,000*y = 0, 45 < x, x < 80) &implies ' &and ('-b*x^3 + 12,000*x^2 - 2,560,000*x + 768,000 = 0, 95 <= b, b <= 105)):
out := QuantifierElimination(f, 'output' = 'rootof');
out := (5930 <= a) &and (a <= RootOf(_Z^4 - 1,428,000*_Z^3 + 620,670,000,000*_Z^2 - 7,420,070,585,984,000*_Z + 22,489,217,064,000,000,000, index = real [1])).

5.2 Simulation and Testing

The interval number in \( \varphi_1 \) is \( b = [0.95, 1.05] \). In addition, \( n \) numbers are selected randomly in intervals \([0.95, 1.05]\) to form \( n \) test cases \( (case_1, case_1, \ldots, case_n) \). These \( n \) random parameters are substituted into Formula (45) to form \( n \) polynomial equation systems, the curves of which are represented by the red parts in Figures 7 and 8. Similarly, repeat the procedure for \( \varphi_2 \). The \( n \) numbers are selected randomly in the interval \( \exists \varphi = [0.99, 1.01] \) to form \( n \) test cases \( (case_1, case_2, \ldots, case_n) \). These \( n \) cases are then substituted into Formula (43). Subsequently, the \( n \) curves are drawn, as indicated by the black parts in Figures 7 and 8.
5.2. Simulation and Testing

In Step 3, the quantifier elimination algorithm (QE) is available in Maple2020. How-

ever, it can only handle first-order polynomial logic with integer coefficients. Hence, For-

mula (49) must be multiplied by an appropriate integer before calling the quantifier elim-

ination method to obtain Formula (49) without floating-point numbers. Subsequently, the

f: = &A [x, y], &E [b], '&and'(a*x^5 – 1,200,000*x^4 + 64,000,000*x^3 –

b*x^3 + 12,000*x^2 – 2,560,000*y +

with(SemiAlgebraicSetTools):

with(RegularChains):

R: = PolynomialRing([x, y, a, b]):

In Step 3, the quantifier elimination algorithm (QE) is available in Maple2020. How-

ever, it can only handle first-order polynomial logic with integer coefficients. Hence, For-

mula (49) must be multiplied by an appropriate integer before calling the quantifier elim-

ination method to obtain Formula (49) without floating-point numbers. Subsequently, the

f: = &A [x, y], &E [b], '&and'(a*x^5 – 1,200,000*x^4 + 64,000,000*x^3 –

b*x^3 + 12,000*x^2 – 2,560,000*y +

with(SemiAlgebraicSetTools):

with(RegularChains):

R: = PolynomialRing([x, y, a, b]):

The Maple codes of Step 3 are as follows:

Example showing 10,000 curves of \( \varphi_1 \) and \( \varphi_2 \).

Figure 7. Example showing 100 curves of \( \varphi_1 \) and \( \varphi_2 \).

Figure 8. Example showing 10,000 curves of \( \varphi_1 \) and \( \varphi_2 \).

Figure 9. Example showing 10,000 curves of \( \varphi_1 \) and \( \varphi_2 \) with \( 0.9883 \leq a \leq 1.0112 \).

Figure 7 show that, when \( 45 \leq x \leq 80 \), the black area (Zero(\( \varphi_1 \))) is inside the red area

(Zero(\( \varphi_2 \))). This shows that these interceptors can hit the target in the range of 45 km to

80 km from the point of origin in the horizontal direction, as long as these interceptors that

satisfy the formula \( \varphi_2 \) can be launched sufficiently densely.

Furthermore, according to the result from Maple in step 4 in Section 5.1, we can conclude

that when \( 0.9883 \leq a \leq 1.0112 \), Zero(\( \varphi_1 \)) \( \subseteq \) Zero(\( \varphi_2 \)) holds. The black part of Figure 8 is

the situation with \( \Pi = [0.999, 1.001] \) as Zero(\( \varphi_1 \)). The situation with \( 0.9883 \leq a \leq 1.0112 \) is

shown in Figure 9.

In Figure 9, in the range of 45–80 km in the horizontal direction, the black part almost ran out of the red area. This suggests that the result \( (0.9883 \leq a \leq 1.0112) \) obtained by our method are also complete, as in the example in Section 4.4.

5.3. Comparison

Most reasoning methods based on theorem proofs do not contain error semantics. For polynomial systems with error coefficients, reasoning methods with error semantics have
advantages over most methods without error semantics. The literature [19] is valuable for linear error assertions, but it is invalid for nonlinear PEAs. The method reported in [14] is effective for polynomial systems with a single error variable. However, there are few reports on reasoning methods with multiple error variables on PEAs. Conversely, some scholars have studied fuzzy reasoning to deal with errors [30]. This type of method provides the fuzzy degree of correctness or similar conclusions, which is different from the reasoning method proposed in this article. The proposed reasoning method can provide a deterministic answer (without a fuzzy degree) to establish whether the conclusion is correct or incorrect.

6. Conclusions

The reasoning method between PEAs, with two theorems involved, and their proofs, are proposed in this study. In addition, for non-linear assertions without polynomials, Taylor expansion can be used to expand them into polynomials for approximation. However, the quantifier elimination algorithm based on cylindrical algebraic decomposition (in the field of symbolic computation) is used in our reasoning method. Although the time complexity of our reasoning method will significantly increase with the increase in the number of variables, our reasoning method can provide a reliable and accurate answer without a fuzzy degree for the reasoning between PEAs, which is necessary in safety-critical systems. The reasoning problem of error polynomials with many variables is expected to be approximated by combining numerical calculation methods. This is also the direction of our study in the future.

Author Contributions: Conceptualization, P.W.; methodology, J.W. and N.X.; software, J.X.; formal analysis, N.X.; writing—original draft preparation, P.W.; writing—review and editing, N.X.; funding acquisition, J.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant No. 61772006.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are included within the article.

Acknowledgments: This work was supported by the National Natural Science Foundation of China under Grant No. 61772006, the Promotion Project of Basic Faculties for Young and Middle-aged College Teachers in Guangxi China Grant No. 2021KY0168, the Talent Introduction Project of Guangxi University for Nationalities China Grant No. 2020KJQD05.

Conflicts of Interest: The authors declare no conflict of interest.

References
1. Clarke, E.M.; Grumberg, O.; Peled, D. Model Checking; MIT Press: Cambridge, MA, USA, 1999.
2. Doyen, L.; Frehse, G.; Pappas, G.J. Handbook of Model Checking; Springer International Publishing: New York, NY, USA, 2018.
3. Fitting, M. First-order logic and automated theorem proving. Studia Log. 1998, 61, 300–302.
4. Shankar, N. Combining theorem proving and model checking through symbolic analysis. Int. Conf. Concurr. Theory. 2000, 1877, 1–16.
5. Uribe, T.E. Combinations of model checking and theorem proving. In Proceedings of the International Workshop on Frontiers of Combining Systems, Nancy, France, 20–24 March 2000; Springer: Berlin/Heidelberg, Germany, 2000; Volume 1794, pp. 151–170.
6. D’Silva, V.; Kroening, D.; Weissenbacher, G. A survey of automated techniques for formal software verification. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 2008, 27, 1165–1178. [CrossRef]
7. Sun, T.; Yu, W.A. Formal Verification Framework for Security Issues of Blockchain Smart Contracts. Electronics 2020, 9, 255. [CrossRef]
8. Kaufmann, D. Formal Verification of Multiplier Circuits Using Computer Algebra. Ph.D. Thesis, Johannes Kepler University Linz, Linz, Austria, 2020.
9. Katoen, J.-P. Labelled transition systems. In Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3472, pp. 615–616.

10. Wu, J. Algebraic methods for mechanical theorem proving in many-valued logics. *Chin. J. Comput.* 1996, 10, 773–779.

11. Fu, J.; Wu, J.; Tan, H. A deductive approach towards reasoning about algebraic transition systems. *Math. Probl. Eng.* 2015, 607013. [CrossRef]

12. Platzer, A. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2010.

13. Platzer, A. A differential approach to equational differential invariants. In Proceedings of the International Conference on Interactive Theorem Proving, Princeton, NJ, USA, 13–15 August 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 28–48.

14. Liu, J.; Zhan, N.; Zhao, H. Computing semi-algebraic invariants for polynomial dynamical systems. In Proceedings of the Ninth ACM International Conference on Embedded Software, Taipei, Taiwan, 9–14 October 2011; Chakraborty, S., Jerraya, A., Baruah, S.K., Fischmeister, S., Eds.; ACM: New York, NY, USA, 2011; pp. 97–106.

15. He, H.; Wu, J. A New Approach to Nonlinear Invariants for Hybrid Systems Based on the Citing Instances Method. *Information 2020*, 11, 246. [CrossRef]

16. Rushby, J. Theorem proving for verification. In *Summer School on Modeling and Verification of Parallel Processes*; Springer: Berlin/Heidelberg, Germany, 2000.

17. Chen, Z.; Zhang, T.; Li, Z. Hybrid Control Scheme Consisting of Adaptive and Optimal Controllers for Flexible-Base Flexible-Joint Space Manipulator with Uncertain Parameters. In Proceedings of the 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 26–27 August 2017; pp. 341–345. [CrossRef]

18. Xu, T.; Yu, H.S.; Yu, J.P.; Meng, X.X. Adaptive disturbance attenuation control of two tank liquid level system with uncertain parameters based on port-controlled Hamiltonian. *IEEE Access* 2020, 8, 47384–47392. [CrossRef]

19. Wu, Y.; Xu, H.; Xu, C.; Chen, K. Uncertain multi-attributes decision making method based on interval number with probability distribution weighted operators and stochastic dominance degree. *Knowl. Based Syst.* 2016, 113, 199–209. [CrossRef]

20. Liu, F.; Pedrycz, W.; Zhang, W.-G. Limited rationality and its quantification through the interval number judgments with permutations. *IEEE Trans. Cybern.* 2017, 47, 4025–4037. [CrossRef] [PubMed]

21. Xia, B.; Xiao, R.; Yang, L. Solving parametric semi-algebraic systems. In Proceedings of the 7th Asian Symposium on Computer Mathematics, Seoul, Korea, 8–10 December 2005; Pae, S.-i., Park, H., Eds.; ASCM: Chicago, IL, USA, 2005.

22. Tarski, A. *A Decision Method for Elementary Algebra and Geometry*; University of California Press: Berkeley, CA, USA, 1951.

23. Arnon, D.S.; Collins, G.E.; McCallum, S. Cylindrical algebraic decomposition. I. The basic algorithm. *SIAM J. Comput.* 1984, 13, 865–877. [CrossRef]

24. Mulligan, C.; Davenport, J.; England, M. TheoryGuru: A Mathematica package to apply quantifier elimination technology to economics. In *International Congress on Mathematical Software, South Bend, IN, USA, 24–27 July 2018*; Springer: Cham, Switzerland, 2018. [CrossRef]

25. Tonks, Z. A Poly-algorithmic Quantifier Elimination Package in Maple. In *Maple in Mathematics Education and Research*; Springer International Publishing: New York, NY, USA, 2020; pp. 171–186. [CrossRef]

26. Wang, Y. Error Evaluation Method and Robustness Analysis of Fuzzy Reasoning. Ph.D. Thesis, Zhejiang Sci-Tech University, Hangzhou, China, 2019.