



Project Thesis

# On the Use of Gröbner Bases and Algebraic Methods for the Analysis of Hybrid Automata

Kerstin Bauer  
February 20, 2007

Supervisors:

Prof. Dr. Klaus Schneider

Dr. Raffaella Gentilini

Reactive Systems Group  
Department of Computer Science  
University of Kaiserslautern

---

Vorliegende Projektarbeit wurde von mir selbstständig verfasst. Es wurden keine anderen als die angegebenen Quellen und Hilfsmittel benutzt.

Kaiserslautern, February 20, 2007

Kerstin Bauer

# Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Introduction</b>   | <b>1</b>  |
| <b>2</b> | <b>Preliminaries</b>  | <b>3</b>  |
| 2.1      | Hybrid Automata . . . . .   | 3         |
| 2.2      | Algebraic Hybrid Automata . . . . .   | 4         |
| 2.3      | Basics of Computer Algebra . . . . .  | 5         |
| <b>3</b> | <b>Polynomial Invariant Generation on Algebraic Hybrid Automata</b>                   | <b>11</b> |
| 3.1      | Computing Invariants via Ideal Membership . . . . .                                   | 11        |
| 3.2      | The Algorithm in [SSM04, San05] . . . . .   | 13        |
| 3.2.1    | Step 1: Fixing the Templates . . . . .  | 14        |
| 3.2.2    | Step 2: Deriving Constraints for Template Variables via Ideal<br>Membership . . . . . | 14        |
| 3.2.3    | Step 3: Solving the Constraints . . . . .   | 16        |
| 3.3      | An Application Example . . . . .  | 17        |
| <b>4</b> | <b>Incremental Polynomial Invariant Generation on Algebraic Hybrid Au-<br/>tomata</b> | <b>19</b> |
| 4.1      | Strengthening of the Initial Conditions . . . . .                                     | 19        |
| 4.1.1    | Initiation Rule . . . . .   | 20        |
| 4.1.2    | Discrete Transition . . . . .   | 21        |
| 4.1.3    | Continuous Transition . . . . .   | 22        |
| 4.2      | Incremental Reduction of the Number of Templates Variables . . . . .                  | 24        |
| 4.3      | Incremental Generation of Constraints . . . . .                                       | 27        |
| 4.4      | Analysis of the Solutions . . . . .   | 34        |
| 4.5      | Illustration of the Final Algorithm . . . . .   | 36        |
| <b>5</b> | <b>Summary and Outlook</b>  | <b>43</b> |
|          | <b>Bibliography</b>   | <b>45</b> |



# Chapter 1

## Introduction

Hybrid Automata are proposed in [Hen00] as a formalism for the modeling and the analysis of *hybrid systems*, i.e. systems mixing continuous and discrete dynamics. Hybrid Systems are ubiquitous in many safety-critical fields. Hence, the *safety problem* - i.e. checking if a system respects a number of desirable safety constraints - is recognized as a key problem in the area of hybrid automata. This problem can be formally reduced to the reachability problem, i.e. determining the set of reachable states of a hybrid automaton.

The reachability problem has been proved decidable only for few families of hybrid automata, where either the discrete or the continuous dynamics is required to be utterly simple [HKPV98], while for most interesting classes of hybrid automata the latter problem is undecidable [HKPV98].

One technique to deal with reachability on undecidable hybrid automata is to obtain a finite abstraction which simulates the original system [TK02, RS05]: The underlying simulation relation tells us that if a state is not reachable in the abstracted system, then its property to be unreachable is preserved on the original system [TK02, RS05].

Another approach is to develop techniques for generating invariants of hybrid automata. An invariant is a property  $\psi$ , which holds on all of the reachable states of an automaton. A special type of invariants are inductive invariants or inductive assertions. An inductive assertion is an invariant, which holds initially and which is preserved by all continuous and discrete transitions of the automaton. In [SSM04] and [San05] a method for generating a special class of inductive assertions, so-called algebraic assertions, is proposed. An algebraic assertion for a location  $l$  has the form  $\bigwedge_{i=1}^k p_i(\underline{x}) = 0$ , where the  $p_i$  are polynomials over the system variables  $\underline{x}$ . The key idea of the technique in [SSM04, San05] is to derive for a given *template assertion*, i.e. a parametric polynomial with unknown coefficients and bounded degree, constraints for the coefficients so that the resulting polynomials are ensured to be *inductive invariants*. This method will be described in detail in Chapter 3.

The method proposed in [SSM04, San05] has one main drawback: In the proposed algorithm, the user has to fix a bound on the degrees of the polynomials for the target polynomials in advance. This is a problem, since it is not obvious to see which degrees are useful: too small degrees may only yield the trivial invariant *true*, templates of too high degree require a considerable longer computation time and will possibly lead either to no more invariants as templates of smaller degrees or, even worse, will lead to an

infeasible system of constraints and thus, to no assertion map at all. In this thesis, we therefore present an incremental algorithm for generating invariants, i.e. the effort done to check for polynomials of degree  $n$  will be partly reused when doing computations for higher degrees. We will also discuss further possible improvements of this algorithm. In particular, we propose a strengthening of the initial conditions (i.e. the inductive hypothesis) used for the generation of the constraint systems in [SSM04, San05] and a way to extract more information of the results by the invariant generating algorithm.

# Chapter 2

## Preliminaries

In this chapter we give some basic definitions and fix the notation used in the rest of the thesis.

### 2.1 Hybrid Automata

Hybrid automata are a formal tool introduced in [ACHH93] for the modeling and the analysis of hybrid systems, i.e. systems having mixed continuous/discrete dynamics.

#### Definition 2.1 (Hybrid Automaton)

A hybrid automaton  $H = \langle V, L, T, \theta, D, I, l_0 \rangle$  consists of the following elements:

- $V$  is the set of real-valued system variables,
- $L$  is a finite set of locations,
- $T$  is a finite set of discrete transitions. Every transition  $\tau = (l, l', \phi_\tau) \in T$  consists of the prelocation  $l$ , a postlocation  $l'$  and an assertion  $\phi_\tau$  on the variables  $V \cup V'$  of the pre and postlocation.
- $\theta$  is an assertion which describes the initial condition.
- $D$  is a differential rule, which maps each location  $l$  to an assertion  $D(l)$  over  $V \cup \{\dot{x} \mid x \in V\}$ , which describe the change of the real-valued variables over time.
- $I$  is a map, which maps each location  $l$  to a location invariant  $I(l)$  over  $V$ .
- $l_0$  is the initial location.

**Example 2.1**

Figure 2.1 shows an example of a hybrid automaton (see [Hen00]). Here, a heating system with the two different discrete states **off** and **on** is modeled. When the heating is off, the temperature falls via the differential rule  $\dot{x} = -0.1x$ . When the heating is on, the temperature rises via the differential rule  $\dot{x} = 5 - 0.1x$ . The state **off** can only be left, when the temperature falls below 20 degrees, it must be left, when the temperature would fall below 18 degrees. The state **on** has similar constraints.

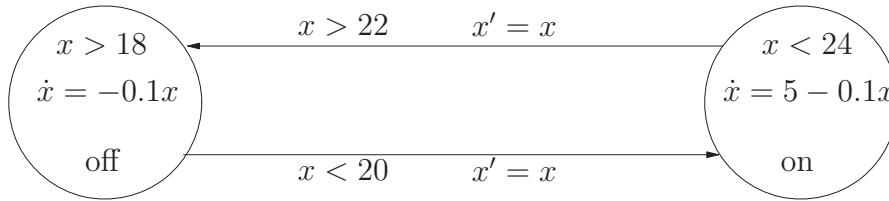


Figure 2.1: heating system [Hen00]

The formal definition of the automaton in Figure 2.1 has the form:  $H = \langle V, L, T, \theta, D, I, l_0 \rangle$

- $V = \{x\}$
- $L = \{\mathbf{off}, \mathbf{on}\}$
- $T = \{(\mathbf{off}, \mathbf{on}, x < 20 \wedge x' = x), (\mathbf{on}, \mathbf{off}, x > 22 \wedge x' = x)\}$
- $\theta = \{x = 20\}$
- $D(\mathbf{off}) = \{\dot{x} = -0.1x\}$  and  $D(\mathbf{on}) = \{\dot{x} = 5 - 0.1x\}$
- $I(\mathbf{off}) = \{x > 18\}$  and  $I(\mathbf{on}) = \{x < 24\}$
- $l_0 = \mathbf{off}$

**2.2 Algebraic Hybrid Automata**

In this thesis, we will only deal with special kinds of hybrid automata, the so-called *algebraic hybrid automata*, see also [SSM04, San05]. In these automata, only algebraic invariants and transition conditions are allowed, i.e. formulas of the form  $\bigwedge_{i=1}^k p_i(\underline{x}) = 0$ , where the  $p_i$  are polynomials in  $\mathbb{R}[\underline{x}]^1$  and also  $\dot{x}_i, x'_i$  are polynomials in  $\mathbb{R}[\underline{x}]$ . More formal:

<sup>1</sup> $\mathbb{R}[\underline{x}]$  is the polynomial ring of the system variables  $\underline{x} = (x_1, \dots, x_n)$  over  $\mathbb{R}$ , see also Definition 2.4

**Definition 2.2 (Algebraic Assertion)**

An algebraic assertion has the form  $\bigwedge_{i=1}^k p_i(\underline{x}) = 0$ , where  $p_i \in \mathbb{R}[\underline{x}]$ . An alternative description for the algebraic assertion is  $\{p_1, \dots, p_k\}$ .

**Definition 2.3 (Algebraic Hybrid Automaton)**

An algebraic hybrid automaton is a hybrid automaton  $H = \langle V, L, T, \theta, D, I, l_0 \rangle$ , where:

- For each discrete transition  $\tau = (l, l', \phi_\tau)$  the relation  $\phi_\tau$  is an algebraic assertion.
- The initial condition  $\theta$  is an algebraic assertion.
- The location invariants  $I(l)$  are algebraic assertions and the differential rules  $D(l)$  have the form  $\dot{x}_i = p(x_1, \dots, x_n)$ , where  $p$  is a polynomial over the variables in  $V$ .

Let us reconsider the heating example of Figure 2.1. Such a hybrid automaton does not represent an algebraic hybrid automaton, since e.g. the location invariant  $x > 18$  for the location **off** is no algebraic assertion. However, the differential rule and the images of the system variable  $x$  of the discrete transitions are algebraic assertions. In this case, ignoring the non-algebraic assertions (here the inequalities) makes the automaton an algebraic one. In this sense, we will still talk about an algebraic automaton, when such inequalities occur, as long as the transition conditions and the initial conditions are algebraic. The non-algebraic conditions will be ignored in this case.

## 2.3 Basics of Computer Algebra

As we can see from the definition of *algebraic hybrid automata* (Definition 2.3), the assertions of an algebraic hybrid automaton in automaton variables  $x_1, \dots, x_n$  have the form  $p(x_1, \dots, x_n) = 0$ , where  $p$  is a polynomial. Our aim is to use computer algebra to analyze the automaton, in particular to verify some specifications. Therefore, we will first list some basics of computer algebra. More detailed information can be found in textbooks like [PG02].

**Definition 2.4 (Polynomial Ring over  $\mathbb{R}$ )**

The polynomial ring over  $\mathbb{R}$  is defined by

$$\mathbb{R}[\underline{x}] := \mathbb{R}[x_1, \dots, x_n] = \left\{ \sum_{\alpha \in A} c_\alpha \cdot x_1^{\alpha_1} \cdot \dots \cdot x_n^{\alpha_n} \mid A \subset \mathbb{N}^n \text{ finite} \right\}$$

Then:

- $\underline{x}^\alpha := x_1^{\alpha_1} \cdot \dots \cdot x_n^{\alpha_n}$  is called a monomial in  $x_1, \dots, x_n$  with exponent vector  $\alpha \in \mathbb{N}^n$ .

- $|\alpha| := \alpha_1 + \dots + \alpha_n$  is the degree of the monomial  $\underline{x}^\alpha$ .
- $Mon(\underline{x}) := \{\underline{x}^\alpha \mid \alpha \in \mathbb{N}\}$ .

Let now  $f = \sum_{\alpha \in A} c_\alpha \cdot x_1^{\alpha_1} \cdot \dots \cdot x_n^{\alpha_n}$  be a polynomial in  $\mathbb{R}[\underline{x}]$ . Then,

- $c_\alpha \underline{x}^\alpha$  is called a term of  $f$  and  $c_\alpha$  the coefficient of  $\underline{x}^\alpha$ .
- $deg(f) := \max\{|\alpha| \mid \alpha \in A\}$  is the degree of  $f$ .

### Definition 2.5 (Ideal)

Let  $\mathbb{R}[\underline{x}]$  be the polynomial ring over  $\mathbb{R}$  and  $I \subseteq \mathbb{R}[\underline{x}]$ . Then  $I$  is called an ideal iff

- $0 \in I$ .
- $\forall a, b \in I : a + b \in I$ .
- $\forall r \in \mathbb{R}[\underline{x}], a \in I : r \cdot a \in I$ .

Notation:  $I \trianglelefteq \mathbb{R}[\underline{x}]$

$\langle p_1, \dots, p_k \rangle = \langle \sum_{i=1}^k q_i \cdot p_i \mid q_i \in \mathbb{R}[\underline{x}] \rangle$  is the ideal generated by  $p_1, \dots, p_k$ , i.e.  $\langle p_1, \dots, p_k \rangle$  is the smallest ideal containing  $\{p_1, \dots, p_k\}$

### Theorem 2.1 (Hilbert's Basis Theorem)

All ideals  $I \trianglelefteq \mathbb{R}[\underline{x}]$  are finitely generated, i.e. for all ideals  $I$  there exist generators  $p_1, \dots, p_k$  in  $\mathbb{R}[\underline{x}]$  s.t.  $I = \langle p_1, \dots, p_k \rangle$ .

From the definition of algebraic hybrid automata, we can see that every reachable state of a location  $l$  satisfies the location conditions  $I(l) = \{p_i(\underline{x}) = 0, i = 1, \dots, k\}$ . In other words, every reachable state of  $l$  lies in the zero-set of the  $p_i, i = 1, \dots, k$ , which is formally denoted as  $V(p_1, \dots, p_k) := \{\underline{x} \in \mathbb{R}^n : p_1(\underline{x}) = \dots = p_k(\underline{x}) = 0\}$ . The latter set can also be described as  $V(I) := \{\underline{x} \in \mathbb{R}^n \mid \forall p \in I : p(\underline{x}) = 0\}$  for the ideal  $I = \langle p_1, \dots, p_k \rangle$ , since all elements of an ideal vanish at a point  $\underline{x} \in \mathbb{R}$  iff all the generators vanish at this point.

Assume now we are given a set of states  $(l, \underline{x})$  encoded by the polynomials  $q_1, \dots, q_j$  and we want to check, if these  $\underline{x}$  satisfy the location conditions  $I(l)$ . With the observation above this problem boils down to checking, whether the polynomials  $q_1, \dots, q_j$  lie in the ideal  $I(l)$  (*Ideal Membership Problem*). This type of problems is solved with *Gröbner Basis Techniques*, which are introduced below.

### Definition 2.6 (Monomial ordering)

A monomial ordering on  $\mathbb{R}[\underline{x}]$  is a relation  $>$  on  $Mon(\underline{x})$  which satisfies

- $>$  is a total ordering on  $Mon(\underline{x})$ .
- $>$  is multiplicative, i.e.  $\underline{x}^\alpha > \underline{x}^\beta$  implies  $\underline{x}^\alpha \cdot \underline{x}^\gamma > \underline{x}^\beta \cdot \underline{x}^\gamma$ .

- $>$  is a well-ordering, i.e.  $\forall \alpha \in \mathbb{N} : \underline{x}^\alpha > 1$ .

Let  $>$  be a monomial ordering and  $f = \sum_{\alpha \in A} c_\alpha \cdot \underline{x}^\alpha \in \mathbb{R}[\underline{x}]$  and let  $\underline{x}^{\alpha_0} = \max\{\underline{x}^\alpha \mid \alpha \in A\}$ .

Then leading monomial, leading term and leading coefficient of  $f$  are defined by

- $lt(f) := c_{\alpha_0} \underline{x}^{\alpha_0}$  is the leading term of  $f$ .
- $lm(f) := \underline{x}^{\alpha_0}$  is the leading monomial of  $f$ .
- $lc(f) := c_{\alpha_0}$  is the leading coefficient of  $f$ .

Moreover, if  $lt(f) = 1$  then the polynomial  $f$  is called monic.

Finally,  $>$  is an elimination ordering on  $Mon(x_1, \dots, x_n, y_1, \dots, y_n)$  w.r.t.  $\underline{x}$  iff  $lm(f) \in \mathbb{R}[\underline{y}] \Rightarrow f \in \mathbb{R}[\underline{y}]$

### Remark 2.1

The following are the most important examples for monomial orderings:

- *Lexicographic order (w.r.t.  $x_1 > x_2 > \dots > x_n$ )*  
 $\underline{x}^\alpha >_{lp} \underline{x}^\beta :\Leftrightarrow \exists s \alpha_1 = \beta_1, \dots, \alpha_{s-1} = \beta_{s-1}, \alpha_s > \beta_s$
- *Graded lexicographic order*  
 $\underline{x}^\alpha >_{Dp} \underline{x}^\beta :\Leftrightarrow$   
 $|\alpha| > |\beta| \text{ or } (|\alpha| = |\beta| \text{ and } \exists s \text{ such that } \alpha_1 = \beta_1, \dots, \alpha_{s-1} = \beta_{s-1}, \alpha_s > \beta_s)$
- *Graded reverse lexicographic order*  
 $\underline{x}^\alpha >_{dp} \underline{x}^\beta :\Leftrightarrow$   
 $|\alpha| > |\beta| \text{ or } (|\alpha| = |\beta| \text{ and } \exists s \text{ such that } \alpha_n = \beta_n, \dots, \alpha_{s+1} = \beta_{s+1}, \alpha_s > \beta_s)$

The lexicographic order is moreover an elimination ordering.

The aim is to define a normal form of a polynomial  $q$  w.r.t. to a given Ideal. Ideally this normal form should be zero iff  $q \in I$ . So we first of all need to define a reduction relation. A canonical way is to define it via multivariate division with remainder. A bit more formal:

### Definition 2.7 (Normal form & Reduction Relation)

Let  $q = \sum c_\alpha \cdot \underline{x}^\alpha$  be a polynomial and  $I = \{p_1, \dots, p_k\} \subset \mathbb{R}[\underline{x}]$ .  $q$  can be reduced by a polynomial  $p \in I$  iff  $lm(p)$  divides a term  $c \cdot t$  of  $f$ . The reduction  $f \xrightarrow{p} f'$  has the form:  $f' = f - \frac{c \cdot t}{lt(p)} p$ . When no reduction with polynomials in  $P$  can be done,  $f$  has a normal form.  $NF_P(f)$  denotes a normal form of  $f$ , i.e.  $f$  is fully reduced by  $P$ .

**Remark 2.2**

The normal form of a polynomial w.r.t. to a given ideal  $I$  is not uniquely determined. Example:

Let  $I = \langle x^2 + x, x^2 - x \rangle$ . Then:

- $x^3 \xrightarrow{x^2+x} -x^2 \xrightarrow{x^2+x} -x$  is in normal form.
- $x^3 \xrightarrow{x^2+x} -x^2 \xrightarrow{x^2-x} +x$  is in normal form.

Changing now the generators yields:  $I = \langle x^2 + x, x^2 - x \rangle = \langle 2x \rangle = \langle x \rangle$ , so the normal form of  $x^3$  w.r.t.  $I = \langle x \rangle$  would obviously be zero.

Taking special generators of the ideal, the so-called Gröbner basis one can assure that the normal form will be unique.

**Definition 2.8 (Leading Ideal)**

Let  $I \trianglelefteq \mathbb{R}[\underline{x}]$ . Then:

The leading ideal of  $I$  wrt to a given ordering  $>$  is defined by

$$L(I) := L_{>}(I) := \{lt(p) \mid p \in I\}$$

i.e.  $L(I)$  is the ideal generated by the leading terms of  $I$ .

**Definition 2.9 (Gröbner Basis)**

A finite subset  $G = \{p_1, \dots, p_k\}$  of  $I \trianglelefteq \mathbb{R}[\underline{x}]$  is called a Gröbner basis for  $I$  w.r.t.  $>$  iff

$$L(I) = \langle lt(p_1), \dots, lt(p_k) \rangle$$

Equivalently: for each  $f \in I \setminus \{0\}$  there is an element  $p \in G$  s.t.  $lt(p) \mid lt(f)$

A Gröbner basis  $G$  is reduced iff

- $0 \notin G$ .
- for  $f, g \in G$ ,  $f \neq g$ :  $lm(f) \not\mid lm(g)$ .
- all elements in  $G$  are monic.
- for all  $f \in G$  the tail  $f - lt(f)$  of  $f$  is reduced wrt  $G$ .

**Theorem 2.2**

Let  $I \trianglelefteq \mathbb{R}[\underline{x}]$ . Then:

- There exists a reduced Gröbner Basis  $G$  of  $I$ .
- $G$  is uniquely determined.
- $\langle G \rangle = I$ .

- The normal form w.r.t.  $G$  is uniquely defined.
- $f \in I \Leftrightarrow NF_G(f) = 0$ .
- Let  $f$  and  $g$  be polynomials. Then:  

$$NF_G(f + g) = NF_G(NF_G(f) + NF_G(g)) = NF_G(f) + NF_G(g).$$

**Proof**

see e.g. [PG02]

**Remark 2.3**

*Gröbner bases depend on the chosen ordering. The ordering also influences the speed of computations, since computing Gröbner bases w.r.t. some orderings is faster than w.r.t. to other orderings. One way to compute the Gröbner basis of an ideal is Buchberger's algorithm, reported below.*

**Algorithm 1** Buchberger's Algorithm

**Require:**  $I = \langle p_1, \dots, p_k \rangle \trianglelefteq \mathbb{R}[\underline{x}]$ ,

a monomial ordering  $>$

**Ensure:**  $G$  being a Gröbner basis of  $I$  wrt  $>$

$G := \langle p_1, \dots, p_k \rangle$

$P := \{(p_i, p_j) \mid 1 \leq i < j \leq k\}$

**while**  $P \neq \emptyset$  **do**

  Choose  $(f, g) \in P$

$P := P \setminus (f, g)$

$\underline{x}^\alpha := lm(f)$ ;  $\underline{x}^\beta := lm(g)$ ;

$\underline{x}^\gamma := lcm(\underline{x}^\alpha, \underline{x}^\beta)$       where  $lcm$  is short form of least common multiple

$spoly(f, g) := lc(g) \cdot \underline{x}^{\gamma-\alpha} \cdot f - lc(f) \cdot \underline{x}^{\gamma-\beta} \cdot g$

$h := NF_G(spoly(f, g))$

**if**  $h \neq 0$  **then**

$P := P \cup \{(h, g) \mid g \in G\}$

$G := G \cup \{h\}$

OUTPUT:  $G$



# Chapter 3

## Polynomial Invariant Generation on Algebraic Hybrid Automata

An invariant for a hybrid automaton is an assertion  $\varphi(\underline{x})$ , which is valid on each reachable state  $\underline{v}$  of the system. Formally:

### Definition 3.1 (Invariant)

An invariant of a hybrid automaton  $H = \langle V, L, T, \Theta, D, I, l_0 \rangle$  at a location  $l$  is an assertion  $\varphi(\underline{x})$ ,  $\underline{x} \in X$  such that for all  $\underline{v} \in \mathbb{R}^n : (l, \underline{v}) \in \text{ReachSet}(H) \Rightarrow \underline{v} \models \varphi$

In this Chapter we review the technique recently proposed in [SSM04, San05] for generating *polynomial invariants* for *algebraic hybrid automata*. The key idea of the technique in [SSM04, San05] is to derive for a given *template assertion*, i.e. a parametric polynomial with unknown coefficients and bounded degree, constraints for the coefficients so that the resulting polynomials are ensured to be *inductive invariants*, i.e. invariants, which hold initially and are preserved by all continuous and discrete transitions of the automaton.

### 3.1 Computing Invariants via Ideal Membership

Inductive invariants are a special form of invariants. Using the notations in [SSM04, San05] inductive invariants can be defined on the base of the notion of *inductive assertion maps*:

### Definition 3.2 (Inductive Assertion Map)

An inductive assertion map  $\eta$  for a hybrid automaton  $H$  is a function that maps each location  $l \in L$  to an invariant assertion  $\eta(l)$  and satisfies the following conditions:

- *Initiation:*

Let  $l_0$  be the initial state with initial condition  $\Theta$ . Then:  $\Theta \models \eta(l_0)$ .

In other words:  $\eta(l_0)$  has to hold initially.

- *Discrete Transition:*

Let  $\tau : (l_i, l_j, \rho)$  be a discrete transition. Then:  $\eta(l_i) \wedge \rho \models \eta(l_j)'$

where  $\eta(l_j)'$  represents the assertion  $\eta(l_j)$  with the current state variables  $x$  replaced by  $x'$

In other words: if  $\eta(l_i)$  holds in  $l_i$  then after the discrete transition  $\tau$  the assertion  $\eta(l_j)$  has to hold.

- *Continuous Transition:*

Let  $l$  be a location of  $H$ . Then:

1. If  $(l, x_2)$  evolves from  $(l, x_1)$  via the given differential rule at  $l$ , then:

$$x_1 \models \eta(l) \Rightarrow x_2 \models \eta(l)$$

2. If  $\eta$  is an assertion of the form  $\eta(l)(x) = 0$ , then:

$$I(l)(x) = 0 \wedge \eta(l)(x) = 0 \models \dot{\eta}(l)(x) = 0$$

In other words: if  $\eta$  holds in a state  $(l, x)$  then it has to hold on the whole continuous path evolving on  $x$  via the given continuous transition.

### Remark 3.1

Obviously every inductive assertion is an invariant. However, not all invariant assertions are inductive. As an example consider the hybrid automaton modeling a bouncing ball, shown in Figure 3.1, where:

- $y$  represents the distance from the floor.
- $v$  represents the velocity.
- $\delta$  represents the time elapsed since the last bound.

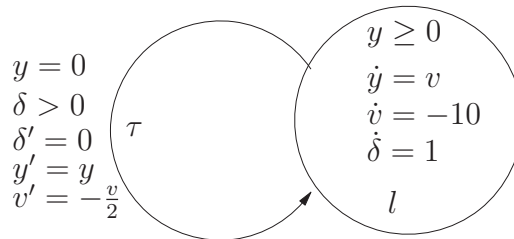


Figure 3.1: Bouncing Ball (see [SSM04, San05])

Assume the automaton has the initial conditions  $\theta = \{y = \delta = 0, v = 16\}$ . With the given initial conditions the assertion  $\phi(l) = v \leq 16$  is obviously an invariant, but it does not fulfill discrete transition, so it is not inductive.

**Definition 3.3 (Template Assertion)**

Let  $A = \{a_1, \dots, a_k\}$  be a set of template variables. Then, a template assertion over the polynomial ring  $\mathbb{R}[\underline{x}]$  is a polynomial of the form  $\sum_{c_\alpha} c_\alpha \cdot \underline{x}^\alpha$  where the  $c_\alpha$  are  $\mathbb{R}$ -linear combinations of the template variables  $A$ .

Given a hybrid automaton  $H = \langle V, L, T, \Theta, D, I, l_0 \rangle$ , let  $\eta$  be a map associating to each location  $l \in L$  of  $H$  a template assertion (over the template variables  $A(l) = \{a_{l,1}, \dots, a_{l,n}\}$ ). Then, in [SSM04, San05] it is shown how the problem of instantiating the template variables of  $\eta$  to get an inductive assertion map is encoded into an ideal membership problem. More precisely, each condition of initiation, discrete transition and continuous transition in Definition 3.2 is mapped to an ideal membership problem, whose solutions allow to obtain a set of constraints for the template variables in  $\eta$ . To give an example, the initiation condition  $\theta \models \eta(l_0)$  is mapped to the ideal membership problem  $\eta(l_0) \in \langle \theta \rangle$  and is solved via Gröbner basis techniques. In particular, standard reduction techniques need to be extended to the notion of template assertions, as illustrated below.

**Definition 3.4 (Template Normal Form NF)**

Let  $f = \sum c_\alpha \cdot \underline{x}^\alpha$  be a template and  $P = \{p_1, \dots, p_k\} \subset \mathbb{R}[\underline{x}]$ .  $f$  can be reduced by a polynomial  $p \in P$  iff  $\text{lm}(p)$  divides a term  $c \cdot t$  of  $f$ . The reduction  $f \xrightarrow{p} f'$  has the form:  $f' = f - \frac{c \cdot t}{\text{lt}(p)} p$ . If no reduction with polynomials in  $P$  can be done,  $f$  has a normal form.  $NF_P(f)$  denotes the normal form of  $f$ , i.e.  $f$  is fully reduced by  $P$ .

**Remark 3.2**

The normal form of a template  $f$  wrt  $P$  is not unique. Uniqueness can be achieved, if a reduced Gröbner basis  $G(P)$  of  $P$  is computed. In particular:  $f \in \langle P \rangle \Leftrightarrow NF_{G(P)} f = 0$ . Thus, the ideal membership problem reduces to compute the normal form wrt the Gröbner basis of the given ideal.

**3.2 The Algorithm in [SSM04, San05]**

Given the above definitions, in this Section we illustrate in detail the algorithm in [SSM04, San05]. Such a procedure makes use of three main steps to derive an algebraic inductive assertion map  $\eta$  for a hybrid automaton  $H$ .

1. In the first step, a template assertion of fixed degree is associated to each location  $l$  in  $H$ .
2. In the second step, a set of constraints on the template variables is derived, which ensures to instantiate  $\eta$  to an inductive assertion map. Deriving the system of constraints boils down to formulate initialization, discrete and continuous transition in Definition 3.2 as ideal membership problems, so that they can be solved with the theory of computer algebra.
3. In the third step, the system of constraint equations from step two is solved.

More precisely, the three steps sketched above are illustrated in the further three subsections.

### 3.2.1 Step 1: Fixing the Templates

For each location  $l \in L$  in the hybrid automaton  $H$ , a degree  $d_l$  is chosen and the corresponding set of template variables  $A_l = \{c_{l,\alpha} \mid |\alpha| \leq d_l\}$  is derived. For maximum of generality, each location should have its own template variables. Given  $l \in L$ , the template associated to  $l$  will then have the form  $\eta(l) = \sum_{|\alpha| \leq d_l} c_{l,\alpha} \cdot \underline{x}^\alpha$

### 3.2.2 Step 2: Deriving Constraints for Template Variables via Ideal Membership

The following rules are used to map each condition in Definition 3.2 to an ideal membership problem, whose solutions lead to a system of constraints over the template variables.

#### Initiation:

Recall from the definition:  $\Theta \models \eta(l_0) = 0$

This is encoded by  $\eta(l) \in \langle \Theta \rangle$  being equivalent to  $NF_\Theta(\eta(l)) = 0$

#### Discrete transition:

Let  $\tau : (l_i, l_j, \rho)$  be a discrete transition.

Recall from the definition:  $\eta(l_i) \wedge \rho \models \eta(l_j)'$

An exact encoding of this condition would mean to compute the Gröbner basis of an ideal generated by  $\eta(l_1) = 0 \wedge \rho$ . The problem here is, that  $\eta(l_1)$  is a template, so that the construction of a Gröbner basis would be too difficult, see therefore [SSM04, San05]. Hence the authors in [SSM04, San05] propose various ways to encode *stronger conditions* which can be handled more easily. The latter conditions are enumerated in the Table 3.1, in particular:

| Name | Condition   | Encoding  |
|------|---|---|
| LC   | $\rho \models \eta(l_2)' = 0$   | $NF_\rho(\eta(l_2)') = 0$                           |
| CS   | $\rho \models \eta(l_1) = \eta(l_2)'$   | $NF_\rho(\eta(l_1) - \eta(l_2)') = 0$               |
| CV   | $\exists \lambda \in \mathbb{R} : \rho \models \eta(l_2)' = \lambda \cdot \eta(l_1)$    | $NF_\rho(\lambda \cdot \eta(l_1) - \eta(l_2)') = 0$ |
| PS   | $\exists q \in \mathbb{R}[\underline{x}] : \rho \models \eta(l_2)' = q \cdot \eta(l_1)$ | $NF_\rho(q \cdot \eta(l_1) - \eta(l_2)') = 0$       |

Table 3.1: Transition conditions for discrete transition

- The *Local Transition condition* (LC) states that the invariant has to hold on the postlocation without using any assumption of the prelocation and the postlocation, i.e. ignoring the fact that the transition has to be feasible that is it has to start in a feasible location  $(l_1, x)$  and land in a feasible state  $(l_2, x')$ . Note, that the local transition condition makes no use of  $\eta(l_1)$ . In figure 3.2 there is shown an example, where discrete transition reduces to a local transition.

In figure 3.2 the condition  $\eta(l_0) = 0$  and  $\eta(l_1) = x$  is an assertion fulfilling LC, since after the discrete transition always  $x = y = 0$  holds.

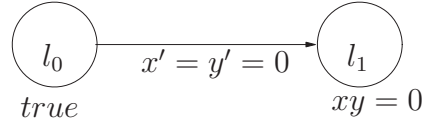


Figure 3.2: Example for LC

- The *Constant Value condition* (CV) states that the value  $\eta(l_1)(x)$  of the prelocation is the same as the value  $\eta(l_2)(x')$  of the postlocation. Together with the fact that  $\eta(l_1)(x) = 0$  because  $\eta(l_1)$  is an assertion you then get that after the transition  $\eta(l_2)(x') = 0$  holds.

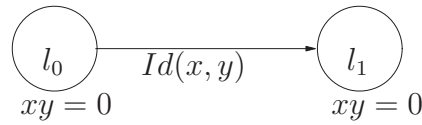


Figure 3.3: Example for CV

In figure 3.3 the condition  $\eta$  is an assertion fulfilling CV for the discrete transition.

- The *Constant Scale condition* (CS) states that the transition may change the value  $\eta(l_1)(x)$  of the prelocation by a constant factor  $\lambda \in \mathbb{R}$ , i.e.  $\lambda \cdot \eta(l_1)(x) = \eta(l_2)(x')$ . Again, together with the fact  $\eta(l_1)(x) = 0$  this yields  $\eta(l_2)(x) = 0$

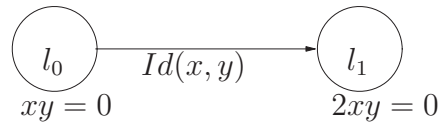


Figure 3.4: Example for CS

In figure 3.4 the condition  $\eta$  together with  $\lambda = 2$  is an assertion fulfilling CS for the discrete transition.

- The *Polynomial Scale condition* (PS) states that the transition may change the value of the assertion by a polynomial factor  $p$ .

Figure 3.5 shows an Example, where  $\eta$  together with  $p = x + 1$  is an assertion fulfilling PS for the discrete transition.

### Continuous transition:

Recall from the definition:  $I(l)(x) = 0 \wedge \eta(l)(x) = 0 \models \dot{\eta}(l)(x) = 0$

Similar to the case of discrete transition an exact encoding of the condition is impracticable (see [SSM04, San05]). Hence the authors of [SSM04, San05] introduce stronger conditions for continuous transition, which are more easily to handle.

- The *constant value* case states that the invariant  $\eta(l)$  remains constant during the continuous path. This means, that the first derivative wrt time gained by

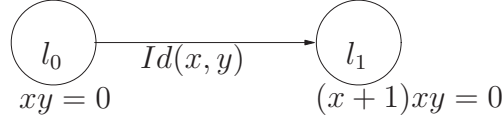


Figure 3.5: Example for PS

| Name | Condition  | Encoding   |
|------|--|--|
| CV   | $I(l) \models \dot{\eta}(l) = 0$   | $NF_{I(l)}(\dot{\eta}(l)) = 0$                         |
| CS   | $\exists \lambda \in \mathbb{R} : I(l) \models \lambda \cdot \eta(l) - \dot{\eta}(l) = 0$    | $NF_{I(l)}(\lambda \cdot \eta(l) - \dot{\eta}(l)) = 0$ |
| PS   | $\exists q \in \mathbb{R}[\underline{x}] : I(l) \models p \cdot \eta(l) - \dot{\eta}(l) = 0$ | $NF_{I(l)}(p \cdot \eta(l) - \dot{\eta}(l)) = 0$       |

Table 3.2: Conditions for continuous transition

the product rule of multivariate analysis  $\dot{\eta}(l) := \sum_i \frac{\partial \eta(l)}{\partial x_i} \cdot \dot{x}_i$  has to be the zero-polynomial. Together with the fact, that  $\eta(l)(x) = 0$  holds at the beginning of the continuous transition, it holds for the whole path.

- The *constant scale* condition (CS) states that the first derivative of  $\eta(l)$  has to be a multiple of  $\eta(l)$ . Together with the fact, that  $\eta(l) = 0$  holds at the beginning of the continuous transition, it then holds during the transition.
- The *polynomial scale* condition (PS) states that the first derivative of  $\eta(l)$  has to be  $\eta(l)$  multiplied with a polynomial factor. Like in the CS case, together with the starting condition  $\eta(l)(x) = 0$ , this condition holds for the whole path evolving from  $x$ .

On the ground of Theorem 3.1, the above mapping of each invariant to an ideal membership problem, i.e. the third columns in Tables 3.1 and 3.2, induce a system of constraints over the template variables.

### Theorem 3.1

Let  $p = \sum_{c_\alpha} c_\alpha \cdot \underline{x}^\alpha \in \mathbb{R}[\underline{x}]$ . Then:  $\forall \underline{x} \in \mathbb{R}^n : p(\underline{x}) = 0 \Leftrightarrow \forall \alpha : c_\alpha = 0$

By Theorem 3.1 it is possible to extract the following equations from the conditions  $NF(\cdot) = 0$ : Let  $0 = NF(\cdot) = \sum_\alpha a_\alpha \cdot \underline{x}^\alpha$ . Then  $\forall \alpha : a_\alpha = 0$ . The above system of equations depends only on the template variables, unknown constants (for the CS case) and unknown polynomials (for the PS case). The complexity of the generated equations depends on the kind of transition conditions adopted, as illustrated in Table 3.3.

### 3.2.3 Step 3: Solving the Constraints

Solving the system of linear / nonlinear equations obtained in step 2 will either lead to no solutions (when the system is infeasible) or to a solution depending on 0 or more parameters  $\lambda_i$ , which are undetermined and can be freely chosen in  $\mathbb{R}$ .

| Condition  | Restriction           | Constraint types     |
|------------|-----------------------|----------------------|
| Initiation | -                     | linear equalities    |
| Transition | Local (LC)            | linear equalities    |
|            | Constant Value (CV)   | linear equalities    |
|            | Constant Scale (CS)   | eigenvalue problems  |
|            | Polynomial Scale (PS) | non-linear algebraic |

Table 3.3: Complexity of the equations for different transition types

**Example 3.1**

The equation  $x = y$  has the solution space  $\{(\lambda, \lambda) | \lambda \in \mathbb{R}\}$

In this case, in order to get rid of the parameters, [SSM04, San05] proposes to set these free parameters equal to 1. This specialization still is a solution, since the  $\lambda_i$  can be freely chosen and the resulting solution is parameter free.

In Chapter 4.4, we will discuss how these free parameters can be used in order to extract more information from the assertion map.

**3.3 An Application Example**

We conclude this chapter by applying the invariant generation algorithm in [SSM04, San05] to a concrete example. Consider again the hybrid automaton in Figure 3.1 which represents a bouncing ball. Assume further the initial condition  $\theta = \{y = \delta = 0, v = 16\}$ .

**Step 1: Fixing the Templates**

Let us take templates of degree 2, i.e.

$$\eta = a + a_y y + a_v v + a_\delta \delta + a_{yy} y^2 + a_{yv} yv + a_{y\delta} y\delta + a_{vv} v^2 + a_{v\delta} v\delta + a_{\delta\delta} \delta^2$$

**Step 2: Deriving Constraints for Template Variables via Ideal Membership**

- Initiation:

$$NF_\theta(\eta) = a + 16a_v + 256a_{vv}$$

- Discrete Transition: LC

$$NF_{\langle y \rangle}(\eta') = a + \frac{a_v}{2} v + \frac{a_{vv}}{4} v^2$$

- Continuous Transition: CV

$$\begin{aligned} NF_{I(l)}(\dot{\eta}) &= a_y v + a_{yy} yv + a_{y\delta} v\delta - 16a_v - 16a_{vv} v - 16a_{yv} y - 16a_{v\delta} \delta \\ &\quad + a_\delta + a_{\delta\delta} \delta + a_{y\delta} y + a_{v\delta} v \\ &= (-16a_v + a_\delta) + (a_{y\delta} - 16a_{yv})y + (a_y - 16a_{vv} + a_{v\delta})v \\ &\quad + (a_{\delta\delta} - 16a_{v\delta})\delta + a_{yy} yv + a_{y\delta} v\delta \end{aligned}$$

**Step 3: Solving the Constraints**

Initiation, discrete and continuous transition lead to the following linear constraints in the template variables:

$$\begin{aligned}
 0 &= a + 16a_v + 256a_{vv} \\
 &= a = \frac{a_v}{2} = \frac{a_{vv}}{4} \\
 &= -16a_v + a_\delta = a_{y\delta} - 16a_{yv} = a_y - 16a_{vv} + a_{v\delta} = a_{yy} = a_{y\delta}
 \end{aligned}$$

Solving this system yields the solution space for  $\eta$ :

$$\forall \lambda \in \mathbb{R} : \lambda \cdot (-y + 5\delta^2 + v\delta) = 0$$

Hence the assertion map  $\eta = -y + 5\delta^2 + v\delta$  is an inductive assertion map

# Chapter 4

## Incremental Polynomial Invariant Generation on Algebraic Hybrid Automata

The main drawback of the method proposed in [SSM04, San05] for the generation of inductive polynomial invariants is, in our view, the need of fixing in advance a bound for the target polynomials. In particular, the user does not know which dimensions of the templates will lead to useful invariants. Templates of too small degree will only lead to the trivial invariant *true*. On the other hand, templates of too high degree won't yield more invariants than templates of smaller degrees. Another problem is that in case where there are no solutions (when the system of equations is infeasible), any search is a waste of time. While the problem of manipulating polynomials of fixed degrees is inherent in the technique in [SSM04, San05], we can at least endow the invariant generation process of an *incrementality* feature, in the sense that the effort done to check for polynomials of degree  $n$  is partly reused when it is necessary/desirable to deal with higher degrees.

In this chapter, we exactly go in the direction of improving the method in [SSM04, San05] by making it incremental. On the way, we will discuss further improvements for the invariants generation of [SSM04, San05]. In particular, in Section 4.1, we propose a strengthening of the initial conditions (i.e. the inductive hypothesis) used for the generation of the constraint systems in [SSM04], [San05].

### 4.1 Strengthening of the Initial Conditions

In Definition 3.2 of inductive invariants, it is important to note that the conditions for initiation, continuous transition and discrete transition only have to hold on reachable states. The invariant generation algorithm proposed in [SSM04, San05] makes only very limited use of this fact. Although the reachability problem itself is undecidable, one can at least state some conditions, which surely have to hold on every reachable state and which can strengthen the hypothesis underlying each inductive rule used for generating invariants in [SSM04, San05]. As an example, consider a discrete transition  $\tau = (l_1, l_2, \rho)$  and the corresponding discrete transition rule  $\eta(l_1) \wedge \rho \models \eta(l_2)$  of Definition 3.2. Since we are interested only in reachable states, we can explicitly state that the transition starts

in a state fulfilling the invariant condition  $I(l_1)$  for  $l_1$  and ends in a state fulfilling the invariant condition  $I(l_2)$  for  $l_2$ . More precisely we can strengthen each inductive rule in Definition 3.2 as described below.

### 4.1.1 Initiation Rule

When one initiates the hybrid automaton with starting values in the initial location  $l_0$ , then obviously the location invariant  $I(l_0)$  also has to be satisfied. This leads to the new initiation rule  $\text{IN}^*$ :  $\theta \wedge I(l_0) \models \eta$  which, encoded as an ideal membership problem yields:  $NF_{\theta \cup I(l_0)}(\eta(l_0)) = 0$ . Using the new initiation rule  $\text{IN}^*$  is especially important, if not all of the hybrid automaton variables are initiated with special values. In the other case the location conditions should be trivially true, otherwise  $\text{IN}^*$  would already lead to a contradiction.

#### Lemma 4.1

*Let  $\eta$  be an inductive assertion map satisfying  $\text{IN}$ . Then it satisfies  $\text{IN}^*$ . The converse is not true.*

#### Proof

Let  $\eta$  satisfy  $\text{IN}$ .

$$\Rightarrow NF_{\theta}(\eta(l_0)) = 0$$

$$\Rightarrow NF_{\theta \cup I(l_0)}(\eta(l_0)) = 0$$

$$\Rightarrow \eta \text{ satisfies } \text{IN}^*$$

Counterexample for the backwards direction:

Consider the automaton shown in Figure 4.1 with initiation conditions  $\theta = \emptyset$ :

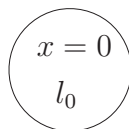


Figure 4.1: Counterexample for Initiation Conditions

$\eta(l_0) = x$  is obviously an inductive assertion map which fulfills  $\text{IN}^*$ .

However,  $\eta(l_0) = x$  does not fulfill  $\text{IN}$ .

q.e.d.

In other words, the initiation condition  $\text{IN}^*$  allows one in principle to obtain a subset of (more precise) overapproximations of reachable sets from the superset of invariants that satisfy  $\text{IN}$ .

#### Example 4.1

*Let us consider the hybrid automaton in Figure 4.2 with initial condition  $\theta = \{y = 1\}$  and term ordering length-lex with  $x > y$ . Let  $\eta(l_0) = a + a_x x + a_y y$*

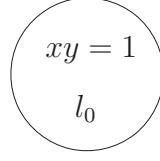


Figure 4.2: Different Initiation Conditions

- *Initiation condition IN:  $\langle \theta \rangle = \langle y - 1 \rangle$*   
 $\Rightarrow NF_{\theta}(\eta(l_0)) = a + a_y + a_x x$   
 $\Rightarrow$  *The corresponding assertion map is  $\forall \lambda \in \mathbb{R} : \eta(l_0) = \lambda \cdot (y - 1)$ .*  
 $\Rightarrow$  *By Theorem 4.12 the invariant  $y = 1$  is found.*  
 $\Rightarrow$  *Every reachable state has to satisfy  $y = 1$ .*
- *Initiation condition IN\*:  $\langle \theta \cup I(l_0) \rangle = \langle x - 1, y - 1 \rangle$*   
 $\Rightarrow NF_{\theta}(\eta(l_0)) = a + a_y + a_x$   
 $\Rightarrow$  *The corresponding assertion map is  $\forall \lambda, \delta \in \mathbb{R} : \eta(l_0) = \lambda \cdot (y - 1) + \delta \cdot (x - 1)$ .*  
 $\Rightarrow$  *By Theorem 4.12, the invariants  $y = 1$  and  $x = 1$  are found.*  
 $\Rightarrow$  *Every reachable state has to satisfy  $y = 1$  and  $x = 1$ .*

$\Rightarrow$  *The overapproximation of reachable states corresponding to the condition IN\* is more precise than the one corresponding to IN.*

### 4.1.2 Discrete Transition

As anticipated in the preamble to this section, a discrete transition  $\tau = (l_1, l_2, \rho)$  can only be triggered, if the conditions  $I(l_1)$  of the prelocation beforehand and  $I(l_2)$  of the postlocation afterwards are satisfied. Assume  $I(l_1) = (p_1(\underline{x}) = 0 \wedge \dots \wedge p_{k_1}(\underline{x}) = 0)$  and denote the invariant conditions for the postlocation  $l_2$  by  $I(l_2)' = (q_1(\underline{x}) = 0 \wedge \dots \wedge q_{k_2}(\underline{x}) = 0)$ . Then, we obtain the following new discrete transition rules:

- **Local Transition (LC\*):**  $I(l_1) = 0 \wedge I'(l_2) \wedge \tau \models \eta(l_2)' = 0$ .  
 Encoded as an ideal membership problem:  $NF_{\rho \cup I(l_1) \cup I'(l_2)}(\eta(l_2)') = 0$ .
- **Constant Value (CV\*):**  $I(l_1) = 0 \wedge I'(l_2) \wedge \tau \models \eta(l_2)' = \eta(l_1)$ .  
 Encoded as an ideal membership problem:  $NF_{\rho \cup I(l_1) \cup I'(l_2)}(\eta(l_2)' - \eta(l_1)) = 0$ .
- **Constant Scale (CS\*):**  $\exists \lambda \in \mathbb{R} : I(l_1) = 0 \wedge I'(l_2) \wedge \tau \models \eta(l_2)' = \lambda \cdot \eta(l_1)$ .  
 Encoded as an ideal membership problem:  $\exists \lambda \in \mathbb{R} : NF_{\rho \cup I(l_1) \cup I'(l_2)}(\eta(l_2)' - \lambda \cdot \eta(l_1)) = 0$ .
- **Polynomial Scale (PS\*):**  $\exists p \in \mathbb{R}[\underline{x}] : I(l_1) = 0 \wedge I'(l_2) \wedge \tau \models \eta(l_2)' = p \cdot \eta(l_1)$ .  
 Encoded as an ideal membership problem:  $\exists p \in \mathbb{R}[\underline{x}] : NF_{\rho \cup I(l_1) \cup I'(l_2)}(\eta(l_2)' - p \cdot \eta(l_1)) = 0$ .

**Lemma 4.2**

Let  $\tau = (l_1, l_2, \rho)$  be a discrete transition. Then:

The conditions  $LC$ ,  $CS$ ,  $CV$  and  $PS$  imply  $LC^*$ ,  $CS^*$ ,  $CV^*$  and  $PS^*$ . The converse is not true.

**Proof**

Let the transition conditions be  $LC$  and  $LC^*$  and let  $\eta$  satisfy  $LC$ .

$$\Rightarrow NF_{\rho}(\eta(l_2)') = 0$$

$$\Rightarrow NF_{I(l_1) \cup \rho \cup I'(l_2)}(\eta(l_2)') = 0$$

$$\Rightarrow LC \text{ implies } LC^*.$$

The other cases are analogous.

Counterexample for the backwards direction (for all transition types):

Consider the automaton in Figure 4.3 with initiation conditions  $\theta = \emptyset$ :

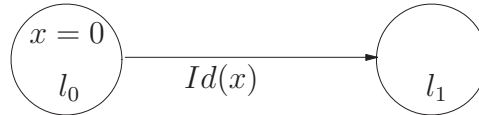


Figure 4.3: Counterexample for Discrete Transition

An inductive assertion map satisfying each of  $LC^*$ ,  $CV^*$ ,  $CS^*$  and  $PS^*$  is:

$$\eta(l_0) = 0 \wedge \eta(l_1) = x$$

However, this inductive assertion map does not fulfill either of  $LC$ ,  $CV$ ,  $CS$  or  $PS$ .  
q.e.d.

**4.1.3 Continuous Transition**

As in the discrete case a continuous transition can be triggered only, if certain circumstances are satisfied. Assume you have a continuous path from  $(l, \underline{x}_1)$  to  $(l, \underline{x}_2)$ . Then, on the whole path the location conditions  $I(l)$  have to be satisfied, i.e. the value of the polynomials in  $I(l)$  is constantly zero and the Lie-Derivative has also to be zero. In other words:

**Lemma 4.3**

Let  $l$  be a location with location conditions  $I(l) = \{p_1(\underline{x}) = 0, \dots, p_k(\underline{x}) = 0\}$ . Let  $(l, \underline{x}_0)$  be reachable. Then, Continuous transition from  $(l, \underline{x}_0)$  is possible, and thus  $\dot{I}(l) := \{\dot{p}_1(\underline{x}) = 0, \dots, \dot{p}_k(\underline{x}) = 0\}$  is satisfied (necessary condition).

**Proof**

Let  $(l, \underline{x}_0)$  be a point, where continuous transition is possible, i.e. there exists a continuous path satisfying  $I(l)$

- $\Rightarrow p_1(\underline{x}) = 0, \dots, p_k(\underline{x}) = 0$  for all  $\underline{x}$  on this path  
 $\Rightarrow$  all the  $p_i$  are constant on the path and thus on the whole path the Lie-derivative of the polynomials is equal to zero. Hence, this holds especially in the starting point of the path.  
 $\Rightarrow \dot{I}(l) := \{\dot{p}_1(\underline{x}) = 0, \dots, \dot{p}_k(\underline{x}) = 0\}$  is also satisfied in  $(l, \underline{x})$ . q.e.d.

By Lemma 4.3, we obtain the following new continuous transition rules:

- Constant Value (CV\*):  $I(l) \wedge \dot{I}(l) \models \dot{\eta}(l) = 0$ .  
 Encoded as an ideal membership problem:  $NF_{I(l) \cup \dot{I}(l)}(\dot{\eta}(l)) = 0$ .
- Constant Scale (CS\*):  $\exists \lambda \in \mathbb{R} : I(l) \wedge \dot{I}(l) \models \dot{\eta}(l) - \lambda \cdot \eta(l) = 0$ .  
 Encoded as an ideal membership problem:  $\exists \lambda \in \mathbb{R} : NF_{I(l) \cup \dot{I}(l)}(\dot{\eta}(l) - \lambda \cdot \eta(l)) = 0$ .
- Polynomial Scale (PS\*):  $\exists p \in \mathbb{R}[x] : I(l) \wedge \dot{I}(l) \models \dot{\eta}(l) - p \cdot \eta(l) = 0$ .  
 Encoded as an ideal membership problem:  $\exists p \in \mathbb{R}[x] : NF_{I(l) \cup \dot{I}(l)}(\dot{\eta}(l) - p \cdot \eta(l)) = 0$ .

#### Lemma 4.4

*Let  $l$  be a location and  $\eta$  be an inductive assertion map. Then, the conditions CV, CS and PS imply CV\*, CS\* and PS\*. The converse is not true.*

#### Proof

Let  $\eta$  satisfy CV for continuous transition  
 $\Rightarrow NF_{I(l)}(\dot{\eta}(l)) = 0$   
 $\Rightarrow NF_{I(l) \cup \dot{I}(l)}(\dot{\eta}(l)) = 0$   
 $\Rightarrow \eta$  satisfies CV\* for continuous transition

The other cases follow analogously.

A counterexample for the backward direction (for all transition types) is the following:

Consider the following automaton in Figure 4.4 with initial condition  $\theta = \{x = 1, y = 0\}$

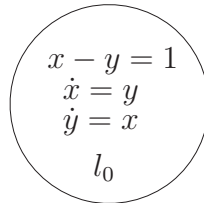


Figure 4.4: Counterexample for Continuous Transition

Then:  $\eta(l_0) = x - 1$  is an inductive assertion map satisfying each of CV\*, CS\* and PS\*, since  $\langle x - y - 1, (x - \dot{y} - 1) \rangle = \langle x - y - 1, y - x \rangle = \langle 1 \rangle = \mathbb{R}$

However,  $\eta(l_0) = x - 1$  does not fulfill any of CV, CS or PS. q.e.d.

## 4.2 Incremental Reduction of the Number of Templates Variables

When deriving new invariants for a location  $l$ , we are only interested in invariants having normal forms regarding to the already existing invariants, i.e. the given invariants  $I(l)$  and all already computed assertions, because adding polynomial combinations of the known invariants gives no new information. This observation together with the new reduction conditions for initiation, discrete and continuous transition can be used to reduce the number of template variables before the beginning of any computation. In particular, Lemma 4.5 and the more trivial results in Lemma 4.6 and Lemma 4.7 allow one to establish Theorem 4.8. According to the latter, it is possible to preliminarily set the value zero to some of the assertion template coefficients for the generating invariant (given a set of precomputed invariants).

### Lemma 4.5

Let  $G = \{p_1, \dots, p_k\}$  be a Gröbner basis for the invariants already holding in a location  $l$  and  $LT(G) := \{lt(p_1), \dots, lt(p_k)\}$ . Then for any further invariant  $q(x) := \sum_{\alpha} c_{\alpha} \cdot x^{\alpha}$  of this location having normal form wrt  $G$  the following is satisfied:  
 $\forall \alpha \in \mathbb{N}^n$  : If any monomial of  $LT(G)$  divides  $x^{\alpha}$  then  $c_{\alpha} = 0$ .

### Proof

Let  $q(x) := \sum_{\alpha} c_{\alpha} \cdot x^{\alpha}$  be a reduced invariant. Assume  $\exists \alpha \in \mathbb{N}^n$  :  $\exists i \in 1, \dots, k$  :  $lt(p_i) | x^{\alpha}$  and  $c_{\alpha} \neq 0$ . Then  $q$  can be reduced via:  $q \rightarrow q - \frac{lc(p_i)}{c_{\alpha}} \cdot \frac{x^{\alpha}}{lt(p_i)} \cdot p_i$ . This is a contradiction to that  $q$  already has normal form wrt  $G$ . q.e.d.

### Lemma 4.6

Let  $l$  be a location with location invariants  $I(l)$  and let  $\eta$  be an inductive assertion map. Then, for the different transition conditions:

- $CV^*$ :  $NF_{I(l) \cup I(l)}(\dot{\eta}(l)) = NF_{I(l) \cup I(l)}(\dot{NF}_{I(l)}(\eta(l)))$
- $CS^*$ :  $NF_{I(l) \cup I(l)}(\dot{\eta}(l) - \lambda \cdot \eta(l)) = NF_{I(l) \cup I(l)}(\dot{NF}_{I(l)}(\eta(l)) - NF_{I(l)}(\lambda \cdot \eta(l)))$
- $PS^*$ :  $NF_{I(l) \cup I(l)}(\dot{\eta}(l) - p \cdot \eta(l)) = NF_{I(l) \cup I(l)}(\dot{NF}_{I(l)}(\eta(l)) - NF_{I(l)}(p \cdot \eta(l)))$

### Proof

Let the transition condition be  $CV^*$  and let  $I(l) = \langle p_1, \dots, p_k \rangle$ .

$$\begin{aligned} \text{Let } \eta(l) &= NF_{I(l)}(\eta(l)) + \sum_{i=1}^k g_i \cdot p_i \\ \Rightarrow \dot{\eta}(l) &= \dot{NF}_{I(l)}(\eta(l)) + \sum_{i=1}^k g_i \cdot p_i \end{aligned}$$

By product rule:  $(\sum_{i=1}^k g_i \cdot p_i) = \sum_{i=1}^k g_i \cdot \dot{p}_i + \dot{g}_i \cdot p_i$ . Computing normal forms wrt Gröbner bases is additive, i.e.  $NF(f + g) = NF(f) + NF(g)$   
 $\Rightarrow NF_{I(l) \cup \dot{I}(l)}(\dot{\eta}(l)) = NF_{I(l) \cup \dot{I}(l)}(NF_{I(l)}(\eta(l))) + NF_{I(l) \cup \dot{I}(l)}(\sum_{i=1}^k g_i \cdot \dot{p}_i + \dot{g}_i \cdot p_i) =$   
 $NF_{I(l) \cup \dot{I}(l)}(NF_{I(l)}(\eta(l))) + 0 = NF_{I(l) \cup \dot{I}(l)}(NF_{I(l)}(\eta(l))),$  since  $\dot{p}_i \in \dot{I}(l)$   
 The other two cases follow analogously. q.e.d.

### Lemma 4.7

Let  $\tau = (l, l', \rho)$  be a discrete transition and let  $\eta$  be an inductive assertion map. Then, for the different transition conditions, we have:

- $LC^*$ :  $NF_{I(l') \cup \rho \cup I'(l)}(\eta(l')) = NF_{I(l') \cup \rho \cup I'(l)}(NF_{I'(l)}(\eta(l')))$
- $CV^*$ :  $NF_{I(l') \cup \rho \cup I'(l)}(\eta(l') - \eta(l)) = NF_{I(l') \cup \rho \cup I'(l)}(NF_{I'(l)}(\eta(l') - \eta(l)))$
- $CS^*$ :  $NF_{I(l') \cup \rho \cup I'(l)}(\eta(l') - \lambda \cdot \eta(l)) = NF_{I(l') \cup \rho \cup I'(l)}(NF_{I'(l)}(\eta(l') - \lambda \cdot \eta(l)))$
- $PS^*$ :  $NF_{I(l') \cup \rho \cup I'(l)}(\eta(l') - p \cdot \eta(l)) = NF_{I(l') \cup \rho \cup I'(l)}(NF_{I'(l)}(\eta(l') - p \cdot \eta(l)))$

### Proof

Analogous to the proof of Lemma 4.6. q.e.d.

### Theorem 4.8

Let  $G = \{p_1, \dots, p_k\}$  be a Gröbner basis for the invariants already holding in a location  $l$  and  $LT(G) := \{lt(p_1), \dots, lt(p_k)\}$ . Consider  $A = \{\alpha \mid \exists x^\beta \in LT(G) : x^\beta \mid x^\alpha\}$ . The template assertion maps  $T = \sum_{\alpha} c_\alpha \underline{x}^\alpha$  and  $T' = \sum_{\alpha \notin A} c_\alpha \underline{x}^\alpha$  are equivalent for all combination of the strengthened transition conditions, i.e. they lead to the same set of new invariants for  $l$ .

### Proof

Let  $\eta(l) = \sum_{\alpha} c_\alpha \cdot \underline{x}^\alpha$ , and let  $x^{\hat{\alpha}}$  be a term of  $\eta(l)$  being divided by the leading term of  $p = \sum_{\gamma} p_\gamma \cdot \underline{x}^\gamma$ ,  $p \in G$ . Since the polynomials in  $G$  are polynomials over  $\mathbb{R}$ , w.l.o.g. we can assume that the leading coefficient of  $p$  is equal to 1.  $G$  is a Gröbner basis, so  $NF_{I(l)}(\eta(l)) = NF_{I(l)}(\eta(l) - c_{\hat{\alpha}} \cdot \frac{x^{\hat{\alpha}}}{lt(p)} p)$  (\*).

Let  $\frac{x^{\hat{\alpha}}}{lt(p)} = \underline{x}^\beta$ , then

$$\begin{aligned} \eta(l) - c_{\hat{\alpha}} \cdot \frac{x^{\hat{\alpha}}}{lt(p)} p &= \sum_{\alpha} c_\alpha \cdot \underline{x}^\alpha - c_{\hat{\alpha}} \cdot \underline{x}^\beta \cdot \sum_{\gamma} p_\gamma \cdot \underline{x}^\gamma \\ &= \sum_{\alpha} c_\alpha \cdot \underline{x}^\alpha - c_{\hat{\alpha}} \cdot \sum_{\gamma} p_\gamma \cdot \underline{x}^{\gamma+\beta} \\ &= \sum_{\alpha} c_\alpha \cdot \underline{x}^\alpha - \sum_{\alpha} p'_\alpha \cdot \underline{x}^\alpha \end{aligned}$$

where in the polynomial  $p$  the terms  $p_\gamma \cdot \underline{x}^{\gamma+\beta}$  are renamed to  $p'_\alpha \cdot \underline{x}^\alpha$  according to  $p'_\alpha = c_{\hat{\alpha}} \cdot p_\beta$  and the coefficients to terms of  $\eta(l)$  not occurring in  $\underline{x}^\beta \cdot p$  are set to zero.

$$\Rightarrow \eta(l) - c_{\hat{\alpha}} \cdot \frac{\underline{x}^{\hat{\alpha}}}{\underline{t}(p)} p = \sum_{\alpha} (c_{\alpha} - p'_{\alpha}) \cdot \underline{x}^{\alpha} = \sum_{\alpha \neq \hat{\alpha}} (c_{\alpha} - p'_{\alpha}) \cdot \underline{x}^{\alpha} \text{ because of the reduction}$$

with  $p$

Renaming the coefficients of  $\eta_{new}(l) := \eta(l) - c_{\hat{\alpha}} \cdot \frac{\underline{x}^{\hat{\alpha}}}{\underline{t}(p)} p$  yields:

$$\eta_{new}(l) = \eta(l) - c_{\hat{\alpha}} \cdot \frac{\underline{x}^{\hat{\alpha}}}{\underline{t}(p)} p = \sum_{\alpha \neq \hat{\alpha}} c'_{\alpha} \underline{x}^{\alpha}$$

Now let us take a look at the three conditions:

- Initiation (if  $l$  is an initial location):

$$NF_{\theta \cup I(l)}(\eta(l)) = NF_{\theta \cup I(l)}(NF_{I(l)}\eta(l)) = NF_{\theta \cup I(l)}(\eta_{new}(l)) \text{ by } (*)$$

- Discrete Transition:

For transition type LC and discrete transition  $\tau = (l', l, \rho)$ :

$$NF_{I(v) \cup \rho \cup I'(l)}(\eta(l')) = NF_{I(v) \cup \rho \cup I'(l)}(NF_{I'(l)}\eta(l')) = NF_{I(v) \cup \rho \cup I'(l)}(\eta_{new}(l')) \text{ by } (*)$$

In the other transition types, one can similarly replace  $\eta(l)'$  by  $\eta_{new}(l)'$ . See Lemma 4.7.

- Continuous Transition:

For transition type CV:

$$NF_{I(l) \cup I(l)}(\dot{\eta}(l)) = NF_{I(l) \cup I(l)}(NF_{I(l)}\dot{\eta}(l)) = NF_{I(l) \cup I(l)}(\dot{\eta}_{new}(l)) \text{ by } (*)$$

In the other transition types, one can similarly replace  $\eta(l)$  by  $\eta_{new}(l)$ . See Lemma 4.6.

Hence, one can replace in each step of the algorithm the polynomial  $\eta(l)$  by  $\eta_{new}(l)$ . Thus, any calculations can be done with  $\eta_{new}(l)$  and afterwards adding the equations  $c'_{\alpha} = c_{\alpha} - p'_{\alpha}$ . Solving the system of coefficient equations with the coefficients  $c'_{\alpha}$  of  $\eta_{new}(l)$  leads to the solution space for  $\eta_{new}(l)$ . Considering the original coefficients of  $\eta(l)$  by  $c'_{\alpha} = c_{\alpha} - p'_{\alpha}$  yields just the addition of  $c_{\hat{\alpha}} \cdot \underline{x}^{\beta} \cdot p$  with arbitrary  $c_{\hat{\alpha}}$ , since in the calculations with  $\eta_{new}(l)$ , no  $c_{\hat{\alpha}}$  occurs and only the equations belonging to the reduction with  $p$  have any  $c_{\hat{\alpha}}$  inside.

Thus, the solution space of the original assertion  $\eta(l)$  has - modulo  $I(l)$  - no more solutions than  $\eta_{new}(l)$ .

Doing this inductively for all reducible terms of  $\eta(l)$  yields the claim. q.e.d.

Lemma 4.5 tells us that setting the coefficients of reducible terms to zero makes sense and Theorem 4.8 tells us, that the algorithm using this fact will - modulo existing invariants - still produce the same output.

### 4.3 Incremental Generation of Constraints

In the previous section, we have seen how to use precomputed invariants to set to 0 some template assertion coefficients of invariants of higher degree. In this section, we further show how to 'recycle' the computations of assertion coefficients along the series of (increasing degree) invariants generated by the algorithm in [SSM04, San05]. We start with some preliminary results, which will be used further.

#### Lemma 4.9

Let  $l$  be a location over the location variables  $\underline{x} = (x_1, \dots, x_n)$ . Let the first derivatives of the  $x_i$  be polynomials in  $\mathbb{R}[\underline{x}]$  and let  $p \in \mathbb{R}[\underline{x}]_{>k}$ . Then,  $\dot{p}[\underline{x}] \in \mathbb{R}[\underline{x}]_{>k-1}$ .

#### Proof

$$\text{By definition: } \dot{p}[\underline{x}] = \sum_{i=1}^k \frac{\partial p}{\partial x_i} \cdot \dot{x}_i$$

$$\text{Let } p = \sum_{k < |\alpha| < \text{deg}(p)} c_\alpha \cdot \underline{x}^\alpha$$

$$\text{Let } c_\alpha \cdot \underline{x}^\alpha = c_\alpha \cdot x_1^{\alpha_1} \dots x_n^{\alpha_n}$$

$$\text{Then } \frac{\partial c_\alpha \cdot \underline{x}^\alpha}{\partial x_i} = \begin{cases} \alpha_i \cdot c_\alpha \cdot x_1^{\alpha_1} \dots x_i^{\alpha_i-1} \dots x_n^{\alpha_n} & \alpha_i \neq 0 \\ 0 & \alpha_i = 0 \end{cases}$$

$$\text{i.e. } \text{deg}\left(\frac{\partial c_\alpha \cdot \underline{x}^\alpha}{\partial x_i}\right) = \begin{cases} \text{deg}(c_\alpha \cdot \underline{x}^\alpha) - 1 & \alpha_i \neq 0 \\ 0 & \alpha_i = 0 \end{cases}$$

$$\Rightarrow \frac{\partial c_\alpha \cdot \underline{x}^\alpha}{\partial x_i} \dot{x}_i = 0 \text{ or } \frac{\partial c_\alpha \cdot \underline{x}^\alpha}{\partial x_i} \dot{x}_i \in \mathbb{R}[\underline{x}]_{>k-1}$$

This yields the claim.

q.e.d.

#### Lemma 4.10

Let  $\tau$  be a transition with  $x'_i \in \mathbb{R}[\underline{x}]_{\geq k_i}$  or  $x'_i = 0$ . Let  $p \in \mathbb{R}[\underline{x}]_{\geq k}$ . Then  $p(\underline{x}') \in \mathbb{R}_{\geq k \cdot \min\{k_i\}}$  or  $p(\underline{x}') = 0$ .

#### Proof

$$\text{Let } p = \sum_{\alpha} c_\alpha \cdot \underline{x}^\alpha.$$

Let  $c_\alpha \cdot \underline{x}^\alpha = c_\alpha \cdot x_1^{\alpha_1} \dots x_n^{\alpha_n}$  be a term of  $p$  and let all  $\alpha_i$  be equal to 0 where  $x'_i = 0$ . Otherwise  $c_\alpha \cdot (\underline{x}^\alpha) = 0$  is not of interest.

$$\text{Then: } c_\alpha \cdot (\underline{x}^\alpha)' = c_\alpha \cdot (x_1^{\alpha_1})' \dots (x_n^{\alpha_n})'$$

$$\Rightarrow \text{deg}(c_\alpha \cdot (\underline{x}^\alpha)') \geq \sum_{i=1}^n \alpha_i \cdot k_i \geq \sum_{i=1}^n \alpha_i \cdot \min\{k_i\} = \min\{k_i\} \cdot \sum_{i=1}^n \alpha_i = \min\{k_i\} \cdot |\alpha| = \min\{k_i\} \cdot \text{deg}(c_\alpha \cdot \underline{x}^\alpha)$$

$\Rightarrow$  Either  $p(\underline{x}') = 0$  or  $p \in \mathbb{R}_{k \cdot \min\{k_i\}}$  since every term of  $p(\underline{x}')$  has at least degree  $k \cdot \min\{k_i\}$ .

q.e.d.

**Lemma 4.11**

Let  $f$  and  $p$  be homogeneous polynomials and let  $f$  be reducible by  $p$ . Then, the reduced polynomial  $f'$  is still homogeneous and is either the zero-polynomial or is of the same degree as  $f$ .

**Proof**

Let  $f = \sum_{\alpha} c_{\alpha} \cdot \underline{x}^{\alpha}$  and let  $c_{\hat{\alpha}} \cdot \underline{x}^{\hat{\alpha}}$  be a reducible term of  $f$ .

Then:  $f \xrightarrow{p} f' = f - \frac{c_{\hat{\alpha}} \cdot \underline{x}^{\hat{\alpha}}}{lt(p)} p$

Since  $\frac{c_{\hat{\alpha}} \cdot \underline{x}^{\hat{\alpha}}}{lt(p)}$  consists only of one term,  $\frac{c_{\hat{\alpha}} \cdot \underline{x}^{\hat{\alpha}}}{lt(p)} \cdot p$  is homogeneous and has degree  $(deg(f) - deg(p)) + deg(p) = deg(f)$ . Adding up two homogeneous polynomials of the same degree either yields the zero-polynomial or a polynomial of the same degree.

This yields the claim.

q.e.d.

**Remark 4.1**

With reduction w.r.t. arbitrary non-homogeneous polynomials, the terms of the reduced polynomial can not be predicted.

Given the above results, we provide a general idea of an incremental variant of the algorithm in [SSM04, San05], where 'incremental' refers to the fact, that

1. to determine the coefficients of the templates referring to invariants having degree  $n$ , we make use of computations relative to lower degree templates, and
2. we make use of already obtained invariants of (lower) degree obtained by performing Gröbner basis reductions.

Assume that all the computations for templates of degree  $n$  are available, and templates of degree  $n + 1$  are to be determined. The new templates will have the form

$$\eta_{n+1}(l) = \sum_{|\alpha| \leq n+1} c_{l,\alpha} \cdot \underline{x}^{\alpha} = \sum_{|\alpha| \leq n} c_{l,\alpha} \cdot \underline{x}^{\alpha} + \sum_{|\alpha|=n+1} c_{l,\alpha} \cdot \underline{x}^{\alpha} =: \eta_n(l) + \eta_{n+1,diff}(l) \quad (*)$$

As you can see from (\*), you can divide the new template into two smaller templates; in particular, the template of one degree less and a template where only terms of degree  $n + 1$  occur. For  $\eta_n(l)$ , the polynomials  $\eta'_n(l)$  and  $\dot{\eta}_n(l)$  have been already computed when dealing with lower degrees. It can easily be shown that  $\eta' + \mu' = (\eta + \mu)'$  and  $\dot{\eta} + \dot{\mu} = (\eta + \mu)$  and in equation (\*),  $NF_G(f + g) = NF_G(f) + NF_G(g)$  holds (see Theorem 2.2). Thus, all operations we deal with are additive. Thus, it remains to apply each one of the above operations to the polynomial  $\eta_{n+1,diff}(l)$  and in a second step, we can combine this with the already existent results for  $\eta_n(l)$ .

Now let us look at  $\eta'_{n+1,diff}$  and  $\dot{\eta}_{n+1,diff}$ . The template  $\eta_{n+1,diff}$  is by definition homogeneous of degree  $n + 1$ . Hence, this yields that  $\dot{\eta}_{n+1,diff} \in \mathbb{R}[\underline{x}]_{>n-1}$ , i.e. no coefficient equation of terms of degree  $\leq n - 1$  will be changed by  $\dot{\eta}_{n+1,diff}$ . The discrete transition is a bit more difficult. If at least one of the polynomials  $x'_i$  has a constant term, all coefficient equations will be changed. In order to illustrate the problem, we will give two simple examples:

**Example 4.2**

- Consider take the following automaton (Figure 4.5). Let the discrete transition type be  $LC^*$  and let  $\eta(l_2) = \sum_i c_i x^i$ .

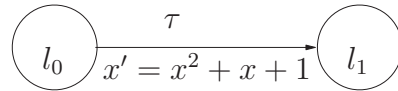


Figure 4.5: Changes due to Discrete Transition (1)

$\Rightarrow \forall n \in \mathbb{N} : \eta'_{n+1,diff} = c_{n+1} \cdot (x')^{n+1} = c_{n+1} \cdot (x^2 + x + 1)^{n+1}$  has terms of every degree  $\leq 2n + 2$ .

$\Rightarrow$  In this example, the transition conditions of the discrete transition  $\tau$  will always change every coefficient equation of  $\eta(l_2)'$ .

- Consider now the slightly modified automaton of Figure 4.6:

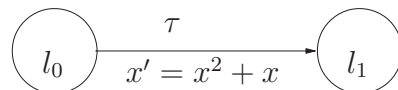


Figure 4.6: Changes due to Discrete Transition (2)

$\Rightarrow \forall n \in \mathbb{N} : \eta'_{n+1,diff} = c_{n+1} \cdot (x')^{n+1} = c_{n+1} \cdot (x^2 + x)^{n+1}$  has only terms of degree  $\geq n + 1$

$\Rightarrow$  When increasing the degree of the template from  $n$  to  $n + 1$ , no coefficient equation for terms of degree  $\leq n$  will be changed for the discrete transition  $\tau$

The second problem is, that the location invariants and guards on the discrete transitions seldom are homogeneous polynomials. Hence, reducing with them will possibly change every coefficient equation. If you want to preserve as many coefficient equations as possible, then you can not automatically do reduction. This will again be illustrated by a simple example:

**Example 4.3**

Consider the following automaton (Figure 4.7) with discrete transition type  $LC^*$ , the lexicographic term ordering with  $x > y$  and let  $\eta(l_2) = \sum_{\alpha} c_{\alpha}(x, y)^{\alpha}$ .

$\Rightarrow \forall n \in \mathbb{N} : \eta'_{n+1,diff} = x^{n+1}$

$\Rightarrow \forall n \in \mathbb{N} : NF(\eta'_{n+1,diff}) = NF(x^{n+1}) = (y + 1)^{n+1}$

$\Rightarrow$  When doing the reduction in  $\eta'_{n+1,diff}$  there will occur terms of every degree.

Because of these two problems, the hybrid automaton has to satisfy the following assumption:

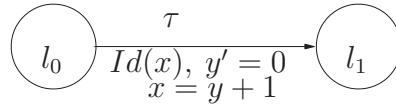


Figure 4.7: Changes due to Computation of Normal Forms

**Assumption 1**

- For all discrete transitions, the polynomials  $x'_i$  are either the zero-polynomials or do not have constant terms.
- All location invariants and transition invariants are homogeneous.

The problem is, that the hybrid automaton seldom has the form claimed in the assumption. We therefore have to modify the automaton in such a way, that (1) the original behavior of the automaton is preserved, and (2) the discrete transitions and given invariants are changed in such a way, that they satisfy Assumption 1. More precisely, after this partial homogenization, the automaton will have the form:

1. For all discrete transitions  $\tau$ :  $x'_i \in \mathbb{R}[x]_{\geq 1}$ , i.e. the  $x'_i$  do not contain a constant term.
2. All location conditions in  $I(l)$ ,  $\dot{I}(l)$  and guards on the discrete transitions are homogeneous polynomials.

After these modifications, the automaton will fulfill exactly the assumptions made for the first approach.

As an example, consider how the automaton of figure 4.8 [SSM04, San05], which models a simple train system, is partly homogenized. In order to achieve an automaton

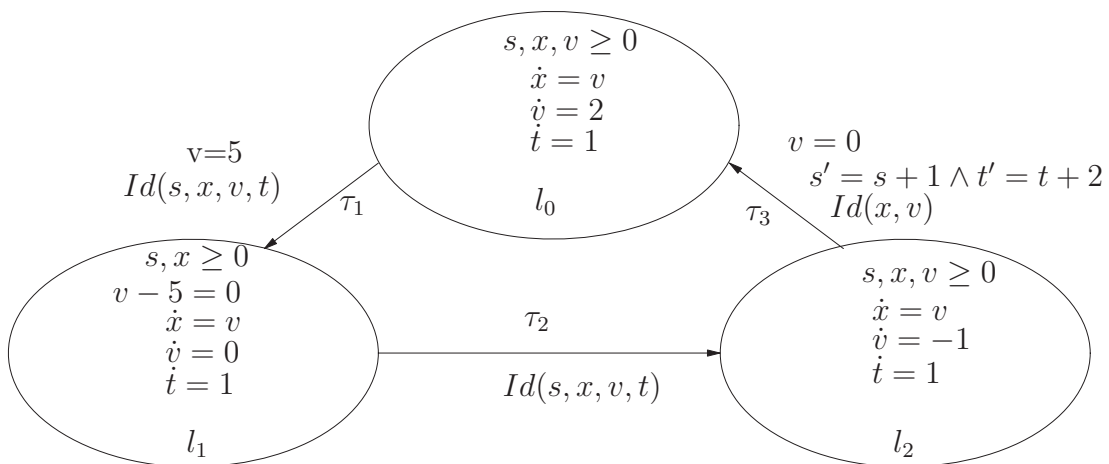


Figure 4.8: Train System (see [SSM04], [San05])

of the desired form, we introduce a slack variable, here denoted by  $y$ .  $y$  will be initialized in the initial state with  $y = 1$ . In each state, the first derivative will be  $\dot{y} = 0$ , i.e.,  $y$

remains constant in each location. The transitions will be expanded with the condition  $y' = y$ . These conditions guarantee that  $y$  will be equally be 1 on all possible paths. Furthermore, the introduction of  $y$  does not change the behavior of the automaton.

In a second step, we change all discrete transition conditions  $x'_i = \dots$  which do not satisfy condition 1 above in the following way:  $x'_i = \sum_{|\alpha| \geq 1} c_\alpha \underline{x}^\alpha + c_0 \mapsto x'_i = \sum_{|\alpha| \geq 1} c_\alpha \underline{x}^\alpha + c_0 \cdot y \in \mathbb{R}[\underline{x}, y]_{>0}$

In the third step, we change the location conditions and the guards on the discrete transitions in such a way that they will be homogenized. Let  $p(\underline{x}) = \sum_{\alpha} c_\alpha \underline{x}^\alpha$  be a polynomial and let  $d$  be the degree of the polynomial. Then the homogenization of  $p$  is  $p^h(\underline{x}) = \sum_{\alpha} c_\alpha \underline{x}^\alpha \cdot y^{d-|\alpha|}$ . For example the homogenization of  $p(x_1, x_2) = x_1^3 x_2 + x_1 x_2^2 + 4x_1$  would be  $p^h(x_1, x_2) = x_1^3 x_2 + x_1 x_2^2 y + 4x_1 y^3$ . If we homogenize the location conditions and the guards on the discrete transitions in such a way, the values of these polynomials do never change, since by construction  $y$  is always equal to 1, and so multiplying summands with powers of  $y$  does not change the values. The same will be done with the conditions  $\dot{I}(l)$  for the continuous transition. If we now construct a Gröbner basis with the homogenized polynomials, the Gröbner basis will also only consist of homogeneous polynomials (see Lemma 4.11).

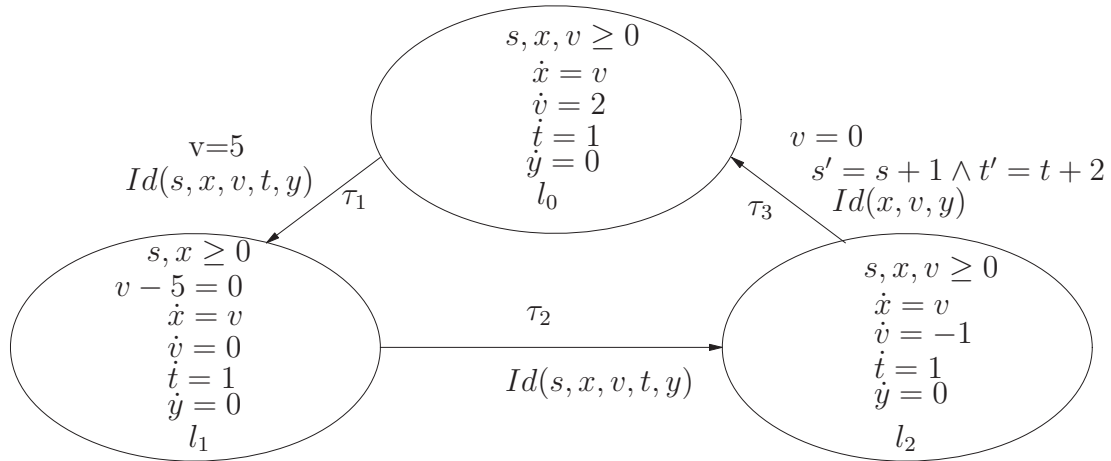


Figure 4.9: Train System after step 1 of the Preprocessing

#### Remark 4.2

*The inductive assertion  $y - 1 = 0$  for all locations  $l$  of the partly homogenized automaton will be found with any combination of  $CV'$ ,  $CS'$  and  $PS'$  for discrete transition and  $CV'$ ,  $CS'$ ,  $PS'$  for continuous transition.*

The formal description of the algorithm is given in Algorithm 2.

After doing this preprocessing, we can now formulate the algorithm for the incremental approach. In addition to the already explained ideas, we can also use Theorem 4.8 in order to reduce the number of template variables.

For the algorithm, we will need the following notations:

---

**Algorithm 2** Computing a partly homogenized automaton
 

---

**Require:** Algebraic Hybrid Automaton  $H$

**Ensure:**  $H'$  satisfying assumption 1

{Introducing the slack variable  $y$ }

**for all** locations  $l$  **do**

  add  $\dot{y} = 0$

**for all** transitions  $\tau$  **do**

  add  $y' = y$

add  $y = 1$  to Initial conditions  $\Theta$

{Changing the discrete transition conditions}

**for all** transitions  $\tau$  **do**

**for all** variables  $x_i$  **do**

    change  $x'_i = \sum_{\alpha \geq 0} c_\alpha \underline{x}^\alpha$  into  $x'_i = \sum_{\alpha > 0} c_\alpha \underline{x}^\alpha + c_{(0, \dots, 0)} \cdot y$

{Homogenizing of the location conditions and transition conditions}

**for all** locations  $l$  **do**

**for all** location invariants  $0 = p(\underline{x})$  **do**

$p := p^h$

**for all** transitions  $\tau$  **do**

**for all** transition conditions  $0 = p(\underline{x})$  **do**

$p := p^h$

---

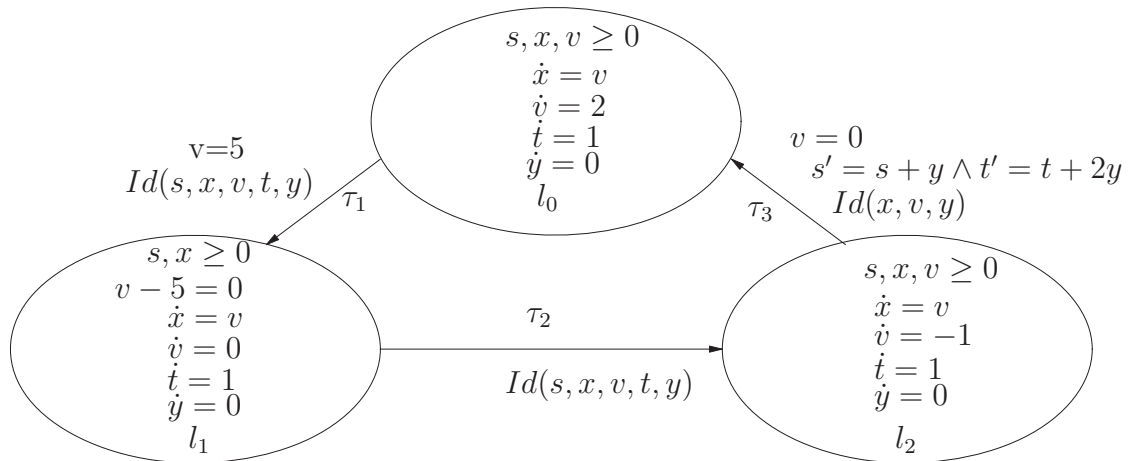


Figure 4.10: Train System after step 2 of the Preprocessing

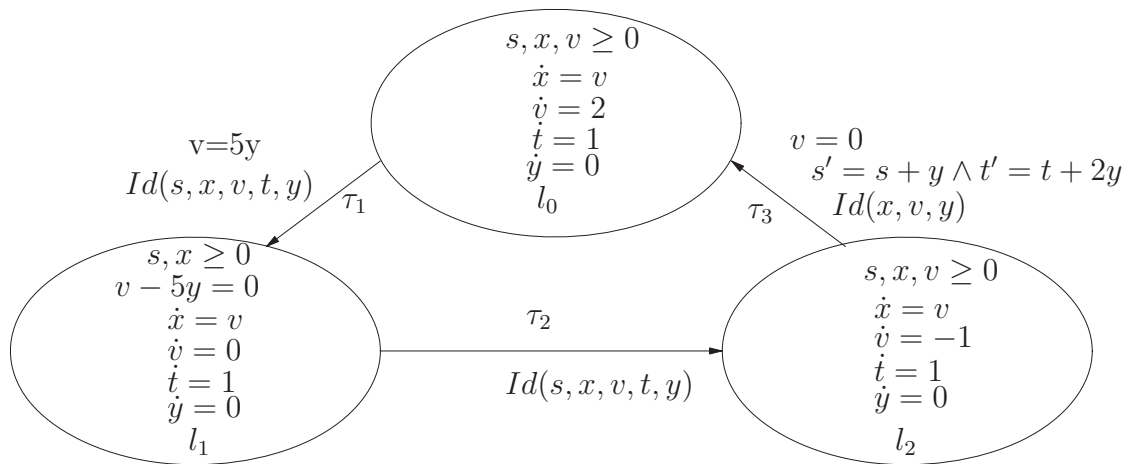


Figure 4.11: Train System after step 3 of the Preprocessing

- $S$  is the Set of coefficient equations, which won't be changed any more,
- $S'$  is a helper set of still changeable coefficient equations,
- $q(l), q(\tau)$  are polynomials for keeping the terms for continuous transition and discrete transition, which can still be changed,
- $pre(\tau)$  and  $post(\tau)$  denote pre and postlocation of a discrete transition  $\tau$ ,
- for  $q(\underline{x}) \in \mathbb{R}[\underline{x}]$ ,  $q_{>i}$  denotes the terms of  $q$  of degree  $> i$  and  $q_{=i}$  the terms of degree  $i$ ,
- $\dot{\eta}^i(l)$  is defined in the following way (depending on the transition condition):
  - CV':  $\dot{\eta}^i(l) := \dot{\eta}_{=i}(l)$
  - CS':  $\dot{\eta}^i(l) := \dot{\eta}_{=i}(l) - \lambda_l \cdot \eta_{=i}$
  - PS':  $\dot{\eta}^i(l) := \dot{\eta}_{=i}(l) - p_l \cdot \eta_{=i}$

Analogously:  $\eta^i(l)'$

- Let  $\tau$  be a discrete transition and  $x'_i \in \mathbb{R}[\underline{x}]_{\geq k_i}$ . Then, as in Lemma 4.10:  
 $k_\tau := \min(k_i)$ .

With these notations, we can now formulate the incremental generation of inductive invariants (Algorithm 3).

## 4.4 Analysis of the Solutions

In this section, we discuss the analysis of the solution space corresponding to a set of coefficient equations. Usually, when the set of constraints is solvable, this set of coefficient equations gives an underestimated system of linear or nonlinear equalities, depending on the chosen transition type for discrete and continuous transition. Assume that you are given the solution space for the coefficients as coefficients depending on parameters  $\underline{\lambda} \in \mathbb{R}^k$ , where  $k$  is the dimension of the solution space. Then, by construction we know that  $\forall \underline{\lambda} \in \mathbb{R}^k \forall \text{location } l : \eta(l)(\underline{\lambda}) = 0$  is an assertion. Here, the  $\eta(l)(\underline{\lambda})$  denotes, that the coefficients of  $\eta(l)$  depend on the vector  $\underline{\lambda}$ . In [SSM04, San05] it is proposed to instantiate the parameters  $\underline{\lambda}$  by  $(1, \dots, 1)$ . Since  $\eta$  is an inductive assertion map for all  $\underline{\lambda} \in \mathbb{R}^k$ , this obviously leads to a feasible inductive assertion map. However, with this approach, a lot of information is lost. Since without losing the property, that  $\eta$  is an inductive assertion map, the value of the parameters  $\underline{\lambda}$  can be assigned any value. This information helps to determine - perhaps non-inductive - assertions from the original parameterized one.

### Theorem 4.12 (Interpretation of Solutions)

*Let  $\eta$  be an inductive assertion map and let  $l$  be a location. Let the coefficients  $c_{l,\alpha}$  of  $\eta(l)$ , short  $c_\alpha$ , be depending on the parameters  $\underline{\lambda} \in \mathbb{R}^k$ , i.e.  $\forall \underline{\lambda} \in \mathbb{R}^k : \eta(l)(\underline{\lambda}) = 0$ . Let  $\eta(l)(\underline{\lambda}) = \sum_{\alpha} c_\alpha(\underline{\lambda}) \cdot \underline{x}^\alpha = \lambda_1 \cdot p_1 + p_2$ , where  $p_1 \in \mathbb{R}[\lambda_1, \dots, \lambda_k][\underline{x}]$  and  $p_2 \in \mathbb{R}[\lambda_2, \dots, \lambda_k][\underline{x}]$ , i.e. divide  $\eta(l)$  in the part depending on  $\lambda_1$  and the part not depending on  $\lambda_1$ . Then:*

$\forall \underline{\lambda} \in \mathbb{R}^k : p_1 = 0$  and  $p_2 = 0$  are algebraic assertions of the location  $l$

### Proof

$\forall \underline{\lambda} \in \mathbb{R}^k : \eta(l)(\underline{\lambda}) = 0$  is an algebraic assertion of  $l$

$\Rightarrow \forall (0, \lambda_2, \dots, \lambda_k) \in \mathbb{R}^k : \eta(l)((0, \lambda_2, \dots, \lambda_k)) = 0$  is an algebraic assertion of  $l$

$\Rightarrow 0 = \eta(l)((0, \lambda_2, \dots, \lambda_k)) = 0 \cdot p_1 + p_2 = p_2$  is an algebraic assertion of  $l$  for all  $\underline{\lambda} \in \mathbb{R}^k$

$\Rightarrow 0 = \eta(l)(\underline{\lambda}) = \lambda_1 \cdot p_1 + p_2 = \lambda_1 \cdot p_1 + 0 = \lambda_1 \cdot p_1$  is an algebraic assertion of  $l$  for all  $\underline{\lambda} \in \mathbb{R}^k$

This yields the claim.

q.e.d.

### Remark 4.3

*Obviously, Theorem 4.12 can be applied inductively on the smaller polynomials  $p_1$  and  $p_2$ . The procedure can also be applied with any other parameter  $\lambda_2, \dots, \lambda_k$ .*

---

**Algorithm 3** Incremental generation of inductive assertions

---

**Require:** Algebraic Hybrid Automaton  $H$ 

type of continuous and discrete transition

Number  $m$  of iteration steps**Ensure:** inductive assertions satisfying the given transition conditions up to degree  $m$ Apply Algorithm 2 to  $H$  $S = \emptyset$ **for all** locations  $l$  and discrete transitions  $\tau$  **do** $p(l) = 0; p(\tau) = 0;$ **for**  $d = 0$  to  $m$  **do** $S' := \emptyset$ **for all** locations  $l$  **do**Set all coefficients of  $\eta$  of degree  $d$  satisfying conditions of Theorem 4.8 equal to zero

{Initiation}

 $S' := S' \cup \{\text{coefficient equations of } NF_{\theta \cup I(l_0)}(\eta(l_0))\}$ 

{Continuous transition}

**for all** locations  $l$  **do** $p(l) := p(l) + NF_{I(l) \cup I(l)}(\dot{\eta}^d(l))$  $S := S \cup \{\text{coefficient equations of } p(l) \text{ of degree } \leq d - 1\}$  $p(l) := p(l)_{\geq d}$  $S' := S' \cup \{\text{coefficient equations of } p(l)\}$ 

{ Discrete transition}

**for all** discrete transitions  $\tau$  **do** $l_1 := \text{pre}(\tau); l_2 := \text{post}(\tau)$  $p(\tau) := p(\tau) + NF_{I(l_1) \cup I(l_2) \cup I(\tau)}(\eta^d(l_2)')$  $S := S \cup \{\text{coefficient equations of } p(\tau) \text{ of degree } < (d + 1) \cdot k_\tau\}$ Solve  $S$  as far as possible (with using the results from former iterations)**if**  $S$  is unsolvable because of contradicting constraints **then**OUTPUT: "No solution for degree  $> d$  possible", STOPsolve  $S \cup S'$ **if**  $S \cup S'$  has solutions **then**actualize the  $I(l)$  and  $\dot{I}(l)$  with the homogenizations of the new solutionsOUTPUT: all solutions of  $S \cup S'$ 

---

## 4.5 Illustration of the Final Algorithm

In this section, we will present example, to illustrate Algorithm 3 as well as Remark 4.2 (the homogenization invariant will be found) and Theorem 4.12. Let us therefore consider the automaton of Figure 4.12 [SSM04, San05], which models the movement of a particle in a magnetic field.

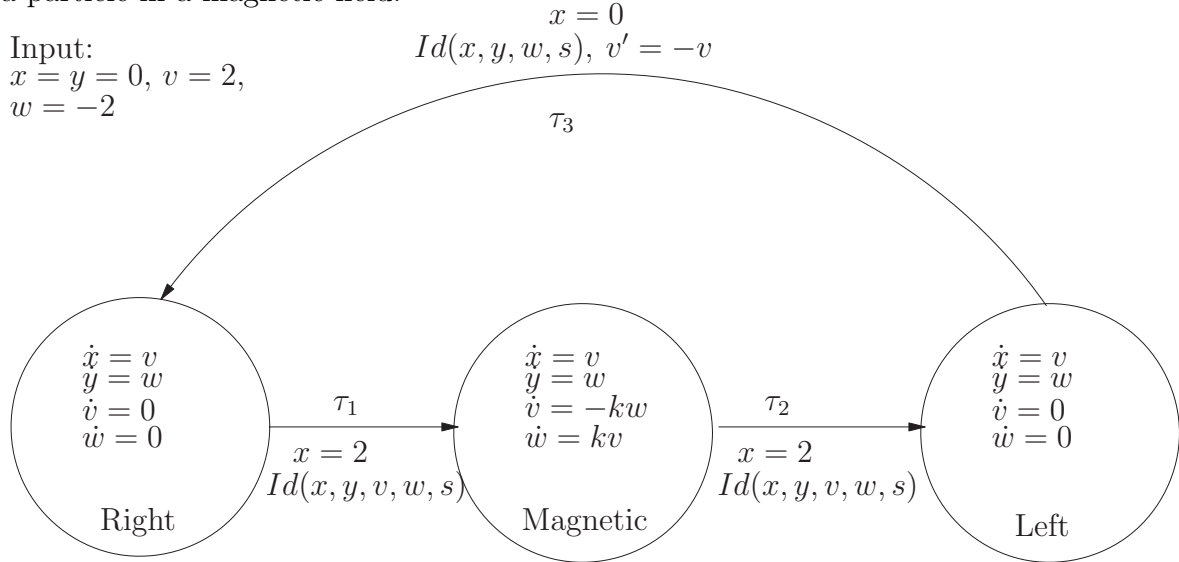


Figure 4.12: Particle in a Magnetic Field

Since the discrete transition conditions of  $\tau_1$  and  $\tau_2$  do not fulfill the conditions of the assumption, the automaton has to be preprocessed by Algorithm 2. Applying this algorithm to the automaton yields the automaton of Figure 4.13.

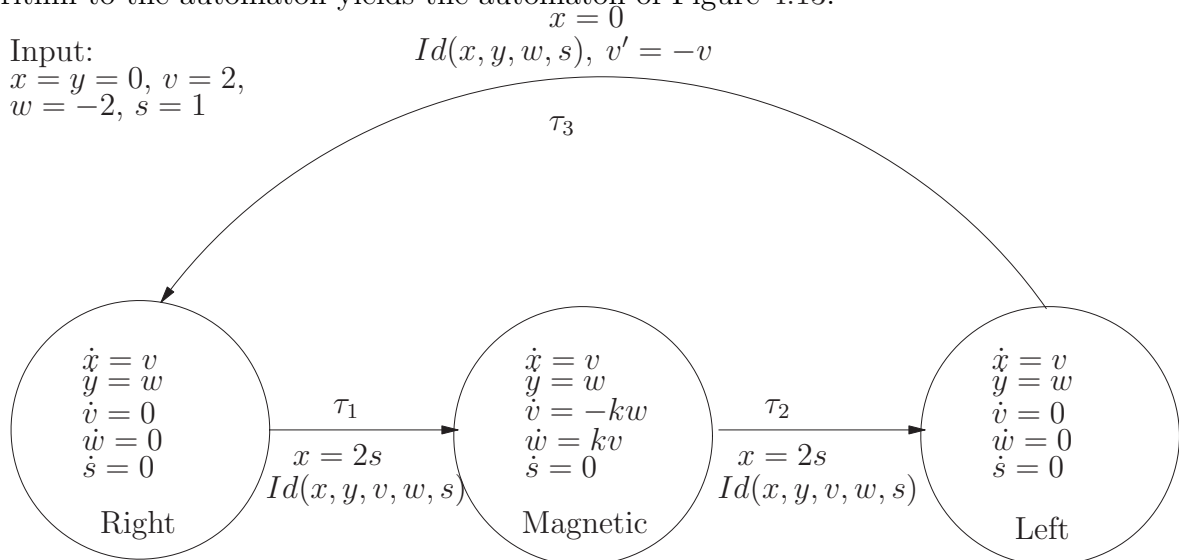


Figure 4.13: Particle in a Magnetic Field: after preprocessing

We will describe the calculations step for step for the degrees 0 up to 2 of the template polynomials. As transition conditions, we choose CV\* for continuous and discrete

transition. The templates will have the following form:

- Location Right:  $\eta(Right) = \sum_{\alpha} a_{\alpha} \underline{x}^{\alpha}$
- Location Magnetic:  $\eta(magnetic) = \sum_{\alpha} b_{\alpha} \underline{x}^{\alpha}$
- Location Left:  $\eta(Left) = \sum_{\alpha} c_{\alpha} \underline{x}^{\alpha}$

The constants  $k_{\tau}$  for the discrete transitions are  $k_{\tau_1} = k_{\tau_2} = k_{\tau_3} = 1$

The term ordering we will use is *length-lex* with  $x > y > v > w > s$ . Since at most one equation is given for each ideal needed for continuous and discrete transition, this polynomial is already the Gröbner basis, so no Gröbner basis computations are required until new invariants are found.

**Degree = 0:**

**Initiation:**  $a = 0$

**Continuous Transition:**

- Location Right:  
 $\dot{\eta}^0(Right) = 0$   
 $\Rightarrow p(Right)_{\geq 0} = 0$
- Location Magnetic:  
 $\dot{\eta}^0(Magnetic) = 0$   
 $\Rightarrow p(Magnetic)_{\geq 0} = 0$
- Location Left:  
 $\dot{\eta}^0(Left) = 0$   
 $\Rightarrow p(Left)_{\geq 0} = 0$

**Discrete Transition:**

- Location Right:  
 $\eta^0(Right)' = a - c \stackrel{!}{=} 0$   
 $\Rightarrow p(\tau_3)_{>0} = 0$  and  $S = S \cup \{a - c\}$
- Location Magnetic:  
 $\eta^0(Magnetic)' = b - a \stackrel{!}{=} 0$   
 $\Rightarrow p(\tau_1)_{>0} = 0$  and  $S = S \cup \{b - a\}$
- Location Left:  
 $\eta^0(Left)' = c - b \stackrel{!}{=} 0$   
 $\Rightarrow p(\tau_2)_{>0} = 0$  and  $S = S \cup \{c - b\}$

$\Rightarrow S = \{a - c, b - a, c - b\}$  and  $S' = \{a\}$

Solving S leads to  $a = b = c = \lambda$  for  $\lambda \in \mathbb{R}$ .

Together with  $S'$  this yields:  $a = b = c = 0$ , i.e. the assertion true for all locations.

**Degree = 1:**

**Initiation:**  $a + 2a_v - 2a_w + a_s = 0$

**Continuous Transition:**

- Location Right:

$$\begin{aligned}\dot{\eta}^1(\text{Right}) &= a_x v + a_y w \stackrel{!}{=} 0 \\ \Rightarrow p(\text{Right})_{\geq 1} &= a_x v + a_y w\end{aligned}$$

- Location Magnetic:

$$\begin{aligned}\dot{\eta}^1(\text{Magnetic}) &= b_x v + b_y w - kb_v w + kb_w v = (b_x + kb_w)v + (b_y - kb_v)w \stackrel{!}{=} 0 \\ \Rightarrow p(\text{Magnetic})_{\geq 1} &= (b_x + kb_w)v + (b_y - kb_v)w\end{aligned}$$

- Location Left:

$$\begin{aligned}\dot{\eta}^1(\text{Left}) &= c_x v + c_y w \stackrel{!}{=} 0 \\ \Rightarrow p(\text{Left})_{\geq 1} &= c_x v + c_y w\end{aligned}$$

**Discrete Transition:**

- Location Right:

$$\begin{aligned}\eta^1(\text{Right})' &= a_y y + a_v v + a_w w + a_s s - (c_y y + c_v v + c_w w + c_s s) \\ &= (a_y - c_y)y + (a_v - c_v)v + (a_w - c_w)w + (a_s - c_s)s \stackrel{!}{=} 0\end{aligned}$$

$$\Rightarrow p(\tau_3)_{>1} = 0 \text{ and } S = S \cup \{a_y - c_y, a_v - c_v, a_w - c_w, a_s - c_s\}$$

- Location Magnetic:

$$\begin{aligned}\eta^1(\text{Magnetic})' &= 2b_x s + b_y y + b_v v + b_w w + b_s s - (2a_x s + a_y y + a_v v + a_w w + a_s s) \\ &= (b_y - a_y)y + (b_v - a_v)v + (b_w - a_w)w + (2b_x + b_s - 2a_x - a_s)s \stackrel{!}{=} 0\end{aligned}$$

$$\Rightarrow p(\tau_1)_{>1} = 0 \text{ and } S = S \cup \{b_y - a_y, b_v - a_v, b_w - a_w, 2b_x + b_s - 2a_x - a_s\}$$

- Location Left:

$$\begin{aligned}\eta^1(\text{Left})' &= 2c_x s + c_y y + c_v v + c_w w + c_s s - (2b_x s + b_y y + b_v v + b_w w + b_s s) \\ &= (c_y - b_y)y + (c_v - b_v)v + (c_w - b_w)w + (2c_x + c_s - 2b_x - b_s)s \stackrel{!}{=} 0\end{aligned}$$

$$\Rightarrow p(\tau_2)_{>1} = 0 \text{ and } S = S \cup \{c_y - b_y, c_v - b_v, c_w - b_w, 2c_x + c_s - 2b_x - b_s\}$$

$\Rightarrow$  So we get the coefficient equations

$$\begin{aligned}S &= \{a - c, b - a, c - b\} \cup \{a_y - c_y, a_v - c_v, a_w - c_w, a_s - c_s\} \\ &\quad \cup \{b_y - a_y, b_v - a_v, b_w - a_w, 2b_x + b_s - 2a_x - a_s\} \\ &\quad \cup \{c_y - b_y, c_v - b_v, c_w - b_w, 2c_x + c_s - 2b_x - b_s\} \\ S' &= \{a + 2a_v - 2a_w + a_s\} \cup \{a_x, a_y\} \cup \{b_x + kb_w, b_y - kb_v\} \cup \{c_x, c_y\}\end{aligned}$$

Solving  $S$  yields together with the results from above:

- $a = b = c = \lambda_1$
- $a_y = b_y = c_y = \lambda_2$

- $a_v = b_v = c_v = \lambda_3$
- $a_w = b_w = c_w = \lambda_4$
- $a_s = b_s, 2b_x + b_s - 2a_x - a_s = 0$  and  $2c_x + c_s - 2b_x - b_s = 0$

Solving now  $S \cup S'$  yields the following inductive assertion map:

- Location Right:  $\forall \lambda_1, \lambda_4 \in \mathbb{R} :$

$$\begin{aligned} \eta(\text{Right}) &= \lambda_4 w + (2\lambda_4 - \lambda_1)s + \lambda_1 \\ &= \lambda_4(w + 2s) + \lambda_1(1 - s) \end{aligned}$$

- Location Magnetic:  $\forall \lambda_1, \lambda_4 \in \mathbb{R} :$

$$\begin{aligned} \eta(\text{Magnetic}) &= -k\lambda_4 x + \lambda_4 w + ((2k + 2)\lambda_4 - \lambda_1)s + \lambda_1 \\ &= \lambda_4(-kx + w + (2k + 2)s) + \lambda_1(1 - s) \end{aligned}$$

- Location Left:  $\forall \lambda_1, \lambda_4 \in \mathbb{R} :$

$$\begin{aligned} \eta(\text{Right}) &= \lambda_4 w + (2\lambda_4 - \lambda_1)s + \lambda_1 \\ &= \lambda_4(w + 2s) + \lambda_1(1 - s) \end{aligned}$$

Using now Theorem 4.12 yields the following assertions for the three locations:

- Location Right:  $w + 2s = 0$  and  $1 - s = 0$
- Location Magnetic:  $-kx + w + (2k + 2)s = 0$  and  $1 - s = 0$
- Location Left:  $w + 2s = 0$  and  $1 - s = 0$

**degree = 2:**

In this step, we can already use the information obtained in the last step:

- Calculating new Gröbner bases for continuous and discrete transitions wrt the new invariants. Note, that the homogenization of  $s - 1 = 0$  does only yield  $0 = 0$ , so it does not appear in any form in the Gröbner bases.
- By Theorem 4.8, we can already set some of the coefficients to zero, i.e.

$$a_{xw} = a_{yw} = a_{vw} = a_{ww} = a_{ws} = 0$$

$$b_{xx} = b_{xy} = b_{xv} = b_{xw} = b_{xs} = 0$$

$$c_{xw} = c_{yw} = c_{vw} = c_{ww} = c_{ws} = 0$$

$$S = S \cup \{a_{xw}, a_{yw}, a_{vw}, a_{ww}, a_{ws}\} \cup \{b_{xx}, b_{xy}, b_{xv}, b_{xw}, b_{xs}\} \cup \{c_{xw}, c_{yw}, c_{vw}, c_{ww}, c_{ws}\}$$

In order to simplify computations, we will already use this fact in the following computations.

**Continuous Transition:**

- Location Right:

$$\begin{aligned} NF(\dot{\eta}^2(Right)) &= NF(2a_{xx}xv + a_{xv}v^2 + a_{xs}vs + 2a_{yy}yw + a_{yx}xw + a_{ys}ws) \\ &= 2a_{xx}xv + a_{xv}v^2 + a_{xs}vs - 4a_{yy}ys - 2a_{yx}xs - 2a_{ys}s^2 \end{aligned}$$

$$\Rightarrow p(Right)_{\geq 1} = a_xv + a_yw + NF(\dot{\eta}^2(Right))$$

- Location Magnetic:

$$\begin{aligned} NF(\dot{\eta}^2(Magnetic)) &= NF(2b_{yy}yw + b_{yv}vw + b_{yw}w^2 + b_{ys}ws \\ &\quad - 2kb_{vv}vw - kb_{yv}yw - kb_{vw}w^2 - kb_{vs}ws \\ &\quad + 2kb_{ww}vw + kb_{vv}v^2 + kb_{yw}yv + kb_{ws}vs) \\ &= (2b_{yy} - kb_{yv})yw + kb_{yw}yv + (b_{yv} - 2kb_{vv} + 2kb_{ww})vw \\ &\quad (b_{yw} - kb_{vw})w^2 + (b_{ys} - kb_{vs})ws + kb_{vv}v^2 + kb_{ws}vs \end{aligned}$$

$$\Rightarrow p(Magnetic)_{\geq 1} = (b_x + kb_w)v + (b_y - kb_v)w + NF(\dot{\eta}^2(Magnetic))$$

- Location Left:

$$\begin{aligned} NF(\dot{\eta}^2(Left)) &= NF(2c_{xx}xv + c_{xv}v^2 + c_{xs}vs + 2c_{yy}yw + c_{yx}xw + c_{ys}ws) \\ &= 2c_{xx}xv + c_{xv}v^2 + c_{xs}vs - 4c_{yy}ys - 2c_{yx}xs - 2c_{ys}s^2 \end{aligned}$$

$$\Rightarrow p(Left)_{\geq 1} = c_xv + c_yw + NF(\dot{\eta}^2(Left))$$

**Discrete Transition:**

- Location Right:

$$\begin{aligned} NF(\eta^2(Right)') &= a_{yy}y^2 + a_{yv}yv + a_{ys}ys + a_{vv}v^2 + a_{vs}s + a_{ss}s^2 \\ &\quad - (c_{yy}y^2 + c_{yv}yv + c_{ys}ys + c_{vv}v^2 + c_{vs}s + c_{ss}s^2) \\ &= (a_{yy} - c_{yy})y^2 + (a_{yv} - c_{yv})yv + (a_{ys} - c_{ys})ys \\ &\quad (a_{vv} - c_{vv})v^2 + (a_{vs} - c_{vs})vs + (a_{ss} - c_{ss})s^2 \end{aligned}$$

$$\Rightarrow p(\tau_3)_{> 2} = 0 \text{ and } S = S \cup \{a_{yy} - c_{yy}, a_{yv} - c_{yv}, a_{ys} - c_{ys}, a_{vv} - c_{vv}, a_{vs} - c_{vs}, a_{ss} - c_{ss}\}$$

- Location Magnetic:

The Gröbner basis of  $\langle w + 2s, x - 2s, -kx + w + (2k + 2)s \rangle$  is  $\{w + 2s, x - 2s\}$

$$\begin{aligned} NF(\eta^2(Magnetic)') &= b_{yy}y^2 + b_{yv}yv - 2b_{yw}ys + b_{ys}ys + \\ &\quad b_{vv}v^2 - 2b_{vw}vs + b_{vs}vs + 4b_{ww}s^2 - 2b_{ws}s^2 + b_{ss}s^2 - \\ &\quad (4a_{xx}s^2 + 2a_{xy}ys + 2a_{xv}vs + 2a_{xs}s^2 + a_{yy}y^2 + a_{yv}yv + \\ &\quad a_{ys}ys + a_{vv}v^2 + a_{vs}vs + a_{ss}s^2) \\ &= (b_{yy} - a_{yy})y^2 + (b_{yv} - a_{yv})yv + (2b_{yw} + b_{ys} - 2a_{xy} - a_{ys})ys \\ &\quad + (b_{vv} - a_{vv})v^2 + (2b_{vw} + b_{vs} - 2a_{xv} - a_{vs})vs \\ &\quad + (4b_{ww} - 2b_{ws} + b_{ss} - 4a_{xx} - 2a_{xs}s^2 - a_{ss})s^2 \end{aligned}$$

$$\Rightarrow p(\tau_1)_{> 2} = 0 \text{ and } S = S \cup \{b_{yy} - a_{yy}, b_{yv} - a_{yv}, 2b_{yw} + b_{ys} - 2a_{xy} - a_{ys}, b_{vv} - a_{vv}, 2b_{vw} + b_{vs} - 2a_{xv} - a_{vs}, 4b_{ww} - 2b_{ws} + b_{ss} - 4a_{xx} - 2a_{xs}s^2 - a_{ss}\}$$

- Location Left:

The Gröbner basis of  $\langle w + 2s, x - 2s, -kx + w + (2k + 2)s \rangle$  is  $\{w + 2s, x - 2s\}$

$$\begin{aligned}
NF(\eta^2(Left)') &= 4c_{xx}s^2 - 2c_{xy}ys - 2c_{xv}vs - 2c_{xs}s^2 + c_{yy}y^2 + c_{yv}yv + c_{ys}ys \\
&\quad + c_{vv}v^2c_{vs}vs + c_{ss}s^2 \\
&\quad (b_{yy}y^2 + b_{yv}yv - 2b_{yw}ys + b_{ys}ys + b_{vv}v^2 - 2b_{vw}vs + b_{vs}vs \\
&\quad + 4b_{ww}s^2 - 2b_{ws}s^2 + b_{ss}s^2) \\
&= (4c_{xx} - 2c_{xs} + c_{ss} - 4b_{ww} + 2b_{ws}s^2 - b_{ss})s^2 \\
&\quad + (-2c_{xy} + c_{ys} + 2b_{yw} - b_{ys})ys + (-2c_{xv} + c_{vs} + 2b_{vw} - b_{vs})vs \\
&\quad + (c_{yy} - b_{yy})y^2 + (c_{yv} - b_{yv})yv + (c_{vv} - b_{vv})v^2
\end{aligned}$$

$$\Rightarrow p(\tau_2)_{>2} = 0 \text{ and } S = S \cup \{4c_{xx} - 2c_{xs} + c_{ss} - 4b_{ww} + 2b_{ws}s^2 - b_{ss}, -2c_{xy} + c_{ys} + 2b_{yw} - b_{ys}, -2c_{xv} + c_{vs} + 2b_{vw} - b_{vs}, c_{yy} - b_{yy}, c_{yv} - b_{yv}, c_{vv} - b_{vv}\}$$

This yields the following for  $S$  and  $S'$  (without the already solved equations):

$$\begin{aligned}
S &= \{a_{yy} - c_{yy}, a_{yv} - c_{yv}, a_{ys} - c_{ys}, a_{vv} - c_{vv}, a_{vs} - c_{vs}, a_{ss} - c_{ss}\} \\
&\cup \{b_{yy} - a_{yy}, b_{yv} - a_{yv}, 2b_{yw} + b_{ys} - 2a_{xy} - a_{ys}, b_{vv} - a_{vv}, 2b_{vw} + b_{vs} - 2a_{xv} - a_{vs}, \\
&\quad 4b_{ww} - 2b_{ws} + b_{ss} - 4a_{xx} - 2a_{xs}s^2 - a_{ss}\} \\
&\cup \{4c_{xx} - 2c_{xs} + c_{ss} - 4b_{ww} + 2b_{ws}s^2 - b_{ss}, -2c_{xy} + c_{ys} + 2b_{yw} - b_{ys}, \\
&\quad -2c_{xv} + c_{vs} + 2b_{vw} - b_{vs}, c_{yy} - b_{yy}, c_{yv} - b_{yv}, c_{vv} - b_{vv}\} \\
S' &= \{a_x, a_y, a_{xx}, a_{xv}, a_{xs}, a_{yy}, a_{yx}, a_{ys}\} \\
&\cup \{c_x, c_y, c_{xx}, c_{xv}, c_{xs}, c_{yy}, c_{yx}, c_{ys}\} \\
&\cup \{b_x + kb_w, b_y - kb_v, 2b_{yy} - kb_{yv}, b_{yw}, \\
&\quad b_{yv} - 2kb_{vv} + 2kb_{ww}, b_{yw} - kb_{vw}, b_{ys} - kb_{vs}, b_{vw}, b_{ws}\}
\end{aligned}$$

Since we do not increase the degree of the templates further in this example, we won't solve  $S$  and  $S'$  separately. Solving  $S \cup S'$  yields the following results for  $\lambda \in \mathbb{R}$ :

- $\eta(Right) = \lambda \cdot (v^2 - 4s^2)$
- $\eta(Magnetic) = \lambda \cdot (v^2 + w^2 - 8s^2)$
- $\eta(Left) = \lambda \cdot (v^2 - 4s^2)$



# Chapter 5

## Summary and Outlook

In this thesis, we dealt with the problem of generating inductive algebraic assertions for hybrid automata. Building upon the theoretical background of [SSM04, San05] we were able to construct an incremental algorithm for the generation of series of algebraic invariants of increasing degrees. In particular,

1. we showed how to use lower degree invariants to set to zero some coefficients in the templates for the generation of higher degree invariants to zero, and
2. we developed a technique, which incrementally makes use of previous computations (for coefficients of lower degree invariants) to synthesize the template coefficients of higher degree invariants.

To maximize the reuse of previous computations in the incremental coefficient generation of point 2 above, we proposed a preprocessing technique consisting of a partly homogenization of the given hybrid automaton.

The main advantage of the new algorithm is, that one needs not to decide beforehand, for which degrees the algorithm should derive invariants, since computations already done for smaller degrees are preserved and reused when doing computations for higher degrees. One disadvantage of the preprocessing technique is that one needs to introduce a new system variable, which means that the number of template variables slightly increases. Another disadvantage is that because of the preprocessing - although the automaton still shows exactly the same behavior - one can not guarantee, that all invariants are found, that could be derived without preprocessing of the automaton.

Another direction to further work could be to extend the idea of dealing with algebraic hybrid automata via ideal membership problems to model checking purposes. In particular this could lead to the determination of underapproximation of reachable states that together with invariants leading to overapproximation of the reachable states would give a better insight in the effective reach-set.



# Bibliography

- [ACHH93] ALUR, R., COURCOUBETIS, C., HENZINGER, T., AND HO, P.-H., *Hybrid Automata: An Algorithmic Approach to the Specification and Verification of Hybrid Systems*, Hybrid Systems (R. Grossmann, A. Nerode, A. Ravn, and H. Rischel, eds.), LNCS, vol. 736, Springer, 1993, pp. 209–229.
- [AHLP00] ALUR, R., HENZINGER, T., LAFFERRIERE, G., AND PAPPAS, G., *Discrete Abstractions of Hybrid Systems*, Proceedings of the IEEE **88** (2000), no. 7, 971–984.
- [Hen00] HENZINGER, T., *The Theory of Hybrid Automata*, Verification of Digital and Hybrid Systems, NATO Advanced Study Institute Series F: Computer and Systems Sciences, vol. 170, Springer, 2000, pp. 265–292.
- [HKPV98] HENZINGER, T., KOPKE, P., PURI, A., AND VARAIYA, P., *What’s decidable about hybrid automata?*, Journal of Computer and System Sciences **57** (1998), no. 1, 94–124.
- [PG02] PFISTER, G., AND GREUEL, G. M., *A Singular Introduction to Commutative Algebra*, Springer, 2002.
- [PJ04] PRAJNA, S., AND JADBABAIE, A., *Safety Verification of Hybrid Systems Using Barrier Certificates*, International Workshop on Hybrid Systems: Computation and Control (HSCC) (Philadelphia, PA, USA) (R. Alur and G. Pappas, eds.), LNCS, vol. 2993, Springer, 2004, pp. 477–492.
- [RS05] RATSCHAN, S., AND SHE, Z., *Safety Verification of Hybrid Systems by Constraint Propagation Based Abstraction Refinement*, Hybrid Systems: Computation and Control (HSCC), vol. LNCS 3414, 2005, pp. 573–589.
- [San05] SANKARANARAYANAN, S., *Mathematical Analysis of Programs*, Ph.D. thesis, Stanford University, 2005.
- [SSM04] SANKARANARAYANAN, S., SIPMA, H., AND MANNA, Z., *Constructing Invariants for Hybrid Systems*, International Workshop on Hybrid Systems: Computation and Control (HSCC) (Philadelphia, PA, USA) (R. Alur and G. Pappas, eds.), LNCS, vol. 2993, Springer, 2004, pp. 539–554.
- [TK02] TIWARI, A., AND KHANNA, G., *Series of Abstraction for Hybrid Automata*, Hybrid Systems: Computation and Control, vol. LNCS 2289, 2002, pp. 465–478.

