A Polynomial-Time Algorithm for an Equivalence Problem which Arises in Hybrid Systems Theory[1]

Bhaskar DasGupta
Department of Computer Science
Rutgers University
Camden, NJ 08102, USA
Email: bhaskar@crab.rutgers.edu

Eduardo D. Sontag
Department of Mathematics
Rutgers University
New Brunswick, NJ 08903, USA
Email: sontag@control.rutgers.edu

Abstract

Piecewise linear (PL) systems provide one systematic approach to discrete-time hybrid systems. They blend switching mechanisms with classical linear components, and model arbitrary interconnections of finite automata and linear systems. Tools from automata theory, logic, and related areas of computer science and finite mathematics are used in the study of PL systems, in conjunction with linear algebra techniques, all in the context of a "PL algebra" formalism. PL systems are of interest as controllers as well as identification models.

Basic questions for any class of systems are those of equivalence, and, in particular, if state spaces are equivalent under a change of variables. This paper studies this state-space equivalence problem for PL systems. The problem was known to be decidable, but its computational complexity was potentially exponential; here it is shown to be solvable in polynomial-time.

1 Introduction

Hybrid systems theory has recently become the focus of increased research, as evidenced for instance by the many papers in the volume [1]. In [7], the second author introduced an approach to discrete-time hybrid systems modeling. The proposed class of *piecewise linear (PL) systems* allows the blending of switching mechanisms with classical linear components, and models arbitrary interconnections of finite automata and linear systems. Tools from automata theory, logic, and related areas of computer science and finite mathematics are used in the study of PL systems, in conjunction with linear algebra techniques. These tools get combined into the logic formalism of the "PL algebra" introduced and developed to some extent in [8].

The paper [7] went on to establish the theoretical possibility of stabilizing rather arbitrary systems using PL controllers (by means of sample-and-hold, since PL systems are only defined in discrete-time). These theorems, though very far from leading to practical stabilization methods, show the potential power of PL systems for control applications. On the other hand, PL systems may also be useful as identification models of the plant itself (by piecewise linear, e.g. linear spline, approximations).

Among the most basic questions which can be asked about any class of systems are those regarding equivalence, such as: given two systems, do they represent the same dynamics under a change of variables? As a preliminary step in answering such a question, one must determine if the state spaces of both systems are isomorphic in an appropriate sense. That is, one needs to know if an invertible change of variables is at all possible. Only later can one ask if the equations are the same. For classical, finite dimensional linear systems, this question is trivial, since only dimensions must match. For finite automata, similarly, the question is also trivial, because the cardinality of the state set is the only property that determines the existence of a relabeling of variables. For other classes of systems, however, the question is not as trivial, and single numbers such as dimensions or cardinalities may not suffice to settle the equivalence problem. For example, if one is dealing with continuous time systems defined by smooth vector fields on manifolds, the natural

[1] This research was supported in part by US Air Force Grant AFOSR-97-0159

changes of variables are smooth transformations, and thus a system whose state space is a unit circle cannot be equivalent to a system whose state space is the real line, even though both systems have "dimension" one. Another illustration, more relevant to the present paper, is provided by systems whose variables are required to remain bounded, for instance, because of saturation effects; a state-space like the unit interval $[-1,1]$ looks very different from the unbounded state-space $\mathbb{R}$, even though both have dimension one.

In this paper, we study this state-space equivalence problem for the PL systems studied in [7], under PL changes of variables. The main result provides a polynomial-time algorithm for the checking of PL isomorphism, assuming that the state space is represented in the specific normal form ("labels") introduced in [8]. Briefly, the paper [8] showed the decidability (recursive computability) of the equivalence problem, but the algorithm that would result from the discussion given there has in principle exponential time complexity. Obviously, having a polynomial time algorithm should have a major impact on future studies of PL systems.

The paper is organized as follows. The class of PL systems is defined in the next section, and general remarks are made there; this section contains no new material, and is included merely for motivational reasons. The following mathematical problem of equivalence is formulated next, and we explain why it is the precise problem that needs to be solved in order to solve a state-space equivalence problem: Consider two-variable polynomials with non-negative integer coefficients, and given two such polynomials, decide if they are equivalent to each other modulo the three equalities $x = 2x+1$, $y^2 = 2y^2 + y$ and $y = x + y + 1$. Our main contribution is to show that this problem can be solved in polynomial time, in both the unit-cost and the logarithmic-cost model.

2 PL Systems

Piecewise linear (or more precisely, piecewise-affine) systems in the sense defined in [7] are discrete-time systems described by equations $x(t+1) = P(x(t), u(t))$ (we write simply "$x^+ = P(x,u)$") for which the transition mapping P is a PL map, that is, there is a decomposition of the state space $\mathcal{X}$ and the input value set $\mathcal{U}$ into finitely many pieces, such that, in each of these pieces, the mapping P is given by an affine function. The decomposition is required to be polyhedral, meaning that each piece is described by a set of linear equalities and inequalities.

For example, linear systems arise in the particular case in which there is just one region. But the PL system paradigm includes many more situations of interest, such as, to take just a few examples, linear systems $x^+ = Ax + B\mathrm{sat}\,(u)$ ($\mathrm{sat}\,(u_1,\ldots,u_m)$ is the vector whose ith component is u_i if $|u_i| \leq 1$ and $\mathrm{sign}\,(u_i)$ otherwise) whose actuators are subject to saturation, switched systems $x^+ = A_i x + B_i u$, where the choice of matrix pair (A_i, B_i) depends on a set of linear constraints on current inputs and states, or systems $x^+ = \mathrm{sat}\,(Ax + Bu)$ for which underflows and overflows in state variables must be taken into account.

As part of the specification of a PL system, one includes explicit constraints on controls and states. Thus, the state space and control-value sets are taken to be subsets $\mathcal{X}$ and $\mathcal{U}$ of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively, which indicate *a priori* restrictions on the allowed ranges of variables. To make the theory stay "piecewise linear", we ask that these sets be definable in terms of a finite number of linear equalities and inequalities. Finite sets are included (n independent linear equalities specify each point), and in this way all finite automata are included as well.

Arbitrary interconnections of linear systems and finite automata can be modeled by PL systems. More precisely, given any finite automaton with state space Q, input-value space T, and transition function $\delta : Q \times T \to Q$, we allow the state q of the automaton to control switching among $|Q|$ possible linear dynamics:

$$\begin{aligned} x^+ &= A_q x + B_q u + c_q \\ q^+ &= \delta(q, h(x,u)) \end{aligned}$$

where $A_1,\ldots,A_{|Q|}$ are matrices of size $n \times n$, $B_1,\ldots,B_{|Q|}$ are matrices of size $n \times m$, and $c_1,\ldots,c_{|Q|}$ are n-vectors, and where $h : \mathbb{R}^n \times \mathbb{R}^m \to T$ is a PL map (representing quantized observations of the linear systems). For a sketch of this reduction to PL systems, as well as for precise definitions of PL systems and more discussion and examples, see the second author's paper in [1].

A more mathematically elegant definition of PL sets and maps can be given, as follows. The *PL subsets* of $\mathbb{R}^n$ are those belonging to the smallest Boolean algebra that contains all the open halfspaces of $\mathbb{R}^n$. A map $f : X \to Y$ between two PL subsets X and Y of $\mathbb{R}^a$ and $\mathbb{R}^b$ respectively, is a *PL map* if its graph is a PL subset of $\mathbb{R}^a \times \mathbb{R}^b$. By a *PL set* one means a PL subset of some $\mathbb{R}^n$. Finally, a PL system is a discrete-time system $x^+ = P(x,u)$ with PL state and input value sets and PL transition P.

Two PL sets X and Y are PL isomorphic if there are a PL maps $f : X \to Y$ and $g : Y \to X$ such that $f \circ g$ and $g \circ f$ both equal the identity, that is, $y = f(x)$ is a bijective piecewise linear map.

A PL isomorphism is nothing else than an operation of the following type: make a finite number of cuts along a set of lines (or segments), apply an affine (linear plus translation) transformation to each piece (not dropping any lower-dimensional pieces), and finally paste it all together. As an example, let us take the interior of the triangle in $\mathbb{R}^2$ obtained as $\mathrm{oc}\,\{(0,0),(1,1),(2,0)\}$, where we are using "oc" to indicate the interior of the convex hull of the corresponding points. (We can also define this set, of course, as the intersection of the three hyperplanes $x_2 > 0$, $x_1 - x_2 > 0$, and $x_1 + x_2 < 2$.) We now show that this triangle is PL isomorphic to the interior of the open square with vertices $(0,0)$, $(1,1)$, $(0,1)$, and $(1,0)$. First we cut along the segment $S_1 = \mathrm{oc}\,\{(1,0),(1,1)\}$, obtaining the union of S_1, S_2, and S_3, where $S_2 = \mathrm{oc}\,\{(0,0),(1,0),(1,1)\}$ and $S_3 = \mathrm{oc}\,\{(1,1),(1,0),(2,0)\}$. Next, we apply the affine transformation

$$Tx \;=\; \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} x - \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

to change S_3 into $S_3' = \mathrm{oc}\,\{(1,1),(0,0),(0,1)\}$. Finally, we apply the affine transformation

$$Tx \;=\; \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x - \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

to change S_1 into the missing diagonal $S_1' = \mathrm{oc}\,\{(0,0),(1,1)\}$, and we glue it all back. See Figure 1.

One of the main results in the paper [8] provided a classification of PL sets under isomorphism. The critical step in this classification is to associate to each PL set X a "label" with the property that two spaces X and Y are isomorphic if and only if their labels are related in a certain manner. (By analogy, two finite-dimensional real vector spaces are linearly isomorphic if and only if their dimensions are the same, i.e., letting the "label" be the dimension, if their labels coincide. But in the PL case, single integers do not suffice as "labels".)

Labels are, by definition, polynomials in two variables x, y with non-negative integer coefficients. We let $\mathcal{S} = \mathbb{N}[x,y]$ denote the collection of all such polynomials. Examples of labels are 1, x, y, x^3, $1 + xy + x^2$, etc. We interpret the sum in $\mathcal{S}$ as union of disjoint sets and the product as Cartesian product of sets, the unit 1 as a one-element set, the variable x as the open interval $(0,1)$, and the variable y as the half-line $(0,+\infty)$. Thus, x^3 is an open cube, and $1 + xy + x^2$ is the union of a point, a disjoint set $(0,1)\times(0,+\infty)$, and a unit square disjoint from both. One may decompose any PL set into a finite union (algebraically, a sum) of objects each of which is linearly isomorphic to a monomial in x and y. (Simplicial decompositions provide a way to do this.) In this manner, a label (nonunique) can be associated to each PL set.

Certain formal equalities are easy to establish. Splitting the interval x as

$$(0,1) \;=\; (0,1/2) \bigcup \{1/2\} \bigcup (1/2,1)\,,$$

and then using affine maps ($t \mapsto 2t$ and $t \to 2t-1$ respectively) to map the first and last interval to x, we obtain "$x = 2x + 1$". On the other hand, the split $y = (0,+\infty) = (0,1)\bigcup\{1\}\bigcup(1,+\infty)$ (and $t \mapsto t-1$ applied to the last set) gives us the identity "$y = x + 1 + y$". Drawing a bisecting line through the first quadrant in $\mathbb{R}^2$ gives "$y^2 = y^2 + y + y^2$" (using, e.g., the linear transformation $(t_1,t_2) \mapsto (t_1 - t_2, t_2)$ to send the lower triangle $\{(t_1,t_2) | t_1 > 0, t_1 > t_2\}$ to y^2). It was shown in [8] that these three identities are enough, in the sense that two sets are isomorphic if and only if their labels can be obtained from each other by using repeatedly these elementary identities. In this paper, we take this result as a starting point.

3 Basic Definitions

Let $\mathcal{S} = \mathbb{N}[x,y]$ denote the collection of all polynomials of two variables with *non-negative* integer coefficients. This set can be seen as a semiring, that is, sums and products obey all the usual rules, except that elements do not have additive inverses, i.e. one cannot "substract" (see e.g. [6] for a survey of several applications of semirings in control theory).

Let $\sigma = \{\sigma_1, \sigma_2, \sigma_3\}$ denote the following set of three equalities:

$$\begin{array}{lcl} \sigma_1 & : & x = 2x + 1 \\ \sigma_2 & : & y^2 = 2y^2 + y \\ \sigma_3 & : & y = x + y + 1 \end{array}$$

Let α_i and β_i denote the left-side and right-side, respectively, of the equality $\sigma_i \in \sigma$ (i.e., α_1 is x, β_1 is $2x+1$, etc.). We let $=_\sigma$ be the semiring congruence generated by these equalities. Explicitly, this is defined as follows.

Definition 3.1 *Given two polynomials $P(x,y)$ and $Q(x,y)$ in $\mathcal{S}$, of same degree n, we say that $P(x,y)$ is equivalent to $Q(x,y)$ modulo the equalities in σ, or just that P is equivalent to Q modulo σ, provided that there is some sequence of polynomials*

$$\begin{aligned} &P(x,y) \;=\; R_0(x,y)\,,\, R_1(x,y)\,,\, R_2(x,y)\,,\, \ldots, \\ &R_t(x,y) \;=\; Q(x,y)\,, \end{aligned}$$

with each $R_i(x,y) \in \mathcal{S}$, such that for every $i > 0$, there are decompositions $R_i(x,y) = A(x,y)C(x,y) + T(x,y)$ and $R_{i-1}(x,y) = B(x,y)C(x,y) + T(x,y)$, where $C(x,y), T(x,y) \in \mathcal{S}$, with the property that $B(x,y)$ is obtained from $A(x,y)$ by applying one of the equalities in σ in either direction (i.e., either $B(x,y) = \alpha_i$ and $A(x,y) = \beta_i$ for some i, or $B(x,y) = \beta_i$ and $A(x,y) = \alpha_i$ for some i).

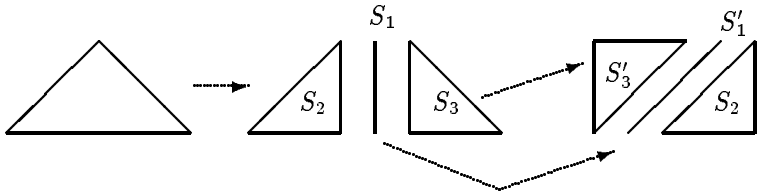

Figure 1: Example: triangle is PL isomorphic to square

We use the notation $P(x,y) =_\sigma Q(x,y)$ to denote two polynomials equivalent as per Definition 3.1, and let $\neq_\sigma$ denote the negation of the relation $=_\sigma$.

Our goal is to develop an efficient algorithm for deciding following question: *Given P and Q, is $P =_\sigma Q$?*

Example: $2x^2 + 2y^2 + y =_\sigma 5x^2 + 4x + y + 1$, since

$$\begin{aligned} 2x^2 + 2y^2 + y &= x^2 + x^2 + 2y^2 + y \\ &=_\sigma x^2 + (2x+1)^2 + 2y^2 + y \\ &=_\sigma 5x^2 + 4x + y + 1 \end{aligned}$$

whereas $y^2 + y \neq_\sigma x^2$.

Notice that $P(x,y) =_\sigma Q(x,y)$ if and only if $Q(x,y) =_\sigma P(x,y)$, since each equality in σ can be applied in either direction, and hence $=_\sigma$ is an equivalence relation on $\mathcal{S}$.

It was shown in [8] that the decision procedure is decidable. Two alternative proofs were sketched there, which were based, respectively, on the algorithms discussed for related semiring problems in the papers [3, 5]. However, no efficient algorithm was known for this procedure, nor was there a proof that the problem is intrinsically hard, i.e. NP-hard. The purpose of this paper is to give a polynomial-time algorithm to decide if $P(x,y) =_\sigma Q(x,y)$. Regarding the precise meaning of polynomial-time computation, there are at least two models of complexity possible. The first, the *unit cost model* of computation, is intended to capture the algebraic complexity of the problem [2]; in that model, each arithmetic and comparison operation on two real numbers is assumed to take unit time. An alternative, the *logarithmic cost model*, is closer to the notion of computation in the usual Turing machine sense (e.g, see [4]); in this case one assumes that each coefficient of the two gives polynomials is an integer with at most B bits, each arithmetic and comparison operation on two B bit integers takes $O(B)$ time, and the time involved in the decision procedure is required to be polynomial on B as well. For convenience, we will just write $O(\alpha)$ time to denote a running time of $O(\alpha)$ in the unit-cost model or a running time of $O(\alpha B)$ in the logarithmic-cost model.

The main theorem of this paper is the following.

Main Theorem. Given any two polynomials $P, Q \in \mathcal{S}$, each of degree at most n, whether $P =_\sigma Q$ or not can be decided in $O(n^2)$ time. Moreover, if $P =_\sigma Q$, then a sequence of valid operations transforming P to Q can be computed in $O(n^2)$ time.

The proof of the main theorem is quite lengthy and complicated, and hence omitted from this paper due to space limitations (the full paper can be obtained from the authors by request). Notice that the time taken by our algorithm for the unit-cost model is optimal within a constant factor in the worst case, since the polynomials have $O(n^2)$ coefficients. Similarly, the time for the logarithmic-cost model is also optimal within a constant factor in the worst case.

In the rest of the paper, we provide a glimpse of the proof for the much simpler case when P and Q are polynomials of one variable x. We assume that both the given polynomials have the same degree, since application of any equality in σ preserves the degree of the polynomial.

4 A Preliminary Result

Let $R = \sum r_{i,j} x^i y^j$ be an intermediate polynomial in a transformation from P to Q (initially, $R = P$).

Definition 4.1 *A* **valid operation** *on the coefficients of a polynomial $R \in \mathcal{S}$ is one of the following operations:*

- *For some $r_{i,j} > 0$, do one of the following:*
  - **Move I(a):** *if $i > 0$, increase both $r_{i,j}$ and $r_{i-1,j}$ by an arbitrary integer $c > 0$.*
  - **Move I(b):** *if $j > 1$, increase both $r_{i,j}$ and $r_{i,j-1}$ by an arbitrary integer $c > 0$.*
  - **Move I(c):** *if $j > 0$, increase both $r_{i,j-1}$ and $r_{i+1,j-1}$ by an arbitrary integer $c > 0$.*
  - **Move I(d):** *if $j > 0$, $r_{i,j-1}, r_{i+1,j-1} \geq c$ for some arbitrary integer $c > 0$, decrease both $r_{i,j-1}$ and $r_{i+1,j-1}$ by c.*
- *For some $r_{i,j} > c > 0$, do one of the following:*

Move II(a): *if $i > 0$ and $r_{i-1,j} \geq c$, decrease both $r_{i,j}$ and $r_{i-1,j}$ by c.*

Move II(b): *if $j > 1$ and $r_{i,j-1} \geq c$, decrease both $r_{i,j}$ and $r_{i,j-1}$ by c.*

After some effort, it can be shown that the problem to decide if $P =_\sigma Q$ can be alternatively formulated as follows.

INSTANCE: Two polynomials $P, Q \in \mathcal{S}$.

QUESTION: Is there a sequence of zero or more valid operations that changes the coefficients of P such that at the end $p_{i,j} = q_{i,j}$ for all i and j?

5 A Few Necessary Conditions for Transformability

In this section, we discuss some necessary and sufficient conditions for transforming the polynomial P to Q. First, we state an easy necessary condition.

Proposition 5.1 *$P =_\sigma Q$ implies $\sum_i \sum_j p_{i,j} = \sum_i \sum_j q_{i,j} \pmod 2$.*

Proof: Every valid operation on a polynomial increases or decreases the total sum of all the coefficients of the polynomial by an even integer. ■

Definition 5.1 *Let $R = \sum r_{i,j} x^i y^j \in \mathbb{Z}[x, y]$ and let n be the degree of the given polynomial P. Define the* **characteristic function** *$\Lambda_n : R \mapsto \mathcal{N}$ to be the following function:*

$$\Lambda_n(R) = \sum_{i=0}^{n} \sum_{j=0}^{n-i} (-1)^{n-j-i} r_{i,j}$$

The idea of the characteristic function is motivated by the *Euler characteristic*[1] function of an abstract simplicial complex on a set (see [10]). Notice that the sign of every coefficient of R is *exactly the opposite* as that of any of its neighbors. The number $\Lambda(R)$ can be computed trivially in $O(n^2)$ time. Note that the Euler characteristic is intimately related to the definition of labels of PL sets and the relations σ; as a matter of fact, the paper [8] (page 200, formula (3.28) and following discussion) explains how the classical theorem on Euler characteristics of polyhedra is a simple consequence of PL set theory.

[1] If f_i is the number of faces of dimension i of an abstract simplicial complex Δ, then the Euler characteristic of Δ is $\sum_{i \geq 0} (-1)^i f_i$

For notational convenience, we will drop the subscript n from Λ_n with the understanding that this is always the degree of P.

Lemma 5.1 *$\Lambda(P) \neq \Lambda(Q)$ implies $P \neq_\sigma Q$.*

Proof: We prove by showing that if P is changed to P' by a single valid operation, then $\Lambda(P) = \Lambda(P')$. This will prove (by induction on the length of the transformation sequence) that the $\Lambda(P)$ must be same as $\Lambda(Q)$ if $P \neq_\sigma Q$.

Consider a valid operation changing P. An inspection of Definition 4.1 shows that a valid operation either changes both $p_{i,j}$ and $p_{i-1,j}$ by the same amount or it changes both $p_{i,j}$ and $p_{i,j-1}$ by the same amount, for some appropriate indices i and j. In the former case, since $p_{i,j}$ and $p_{i-1,j}$ have opposite signs in the expression for $\Lambda(P)$, the value of $\Lambda(P)$ remains unchanged. In the later case also, since $p_{i,j}$ and $p_{i,j-1}$ have opposite signs in the expression for $\Lambda(P)$, the value of $\Lambda(P)$ remains unchanged. ■

Corollary 5.1 *Let P and Q differ in exactly one coefficient. Then, $P \neq_\sigma Q$.*

In fact, the condition stated in Lemma 5.1 can be further strengthened.

Definition 5.2 *For a given polynomial $R = \sum r_{i,j} x^i y^j \in \mathcal{S}$, define $R^1 = \sum_{j=0}^{n} r_{j,0} x^j$ and $R^2 = R - R^1$ (notice that $R^1, R^2 \in \mathcal{S}$).*

As before, let P and Q be the two given polynomials under consideration. Now, we have the following lemma.

Lemma 5.2 *$P =_\sigma Q$ if and only if $\Lambda(P^1) = \Lambda(Q^1)$* **and** *$\Lambda(P^2) = \Lambda(Q^2)$.*

6 The One-Variable Case

In this section we consider the simpler case when both P and Q are polynomials in only one variable x (i.e., $p_{i,j} = q_{i,j} = 0$ for all $j > 0$). This simpler case is important because a technique similar to the one employed for its solution will be applied repeatedly (with appropriate modifications) to solve the original problem. For notational convenience, let us denote $p_{j,0}$ (resp. $q_{j,0}$) simply by p_j (resp. q_j). Notice that $p_n, q_n > 0$ and $\Lambda(P^2 - Q^2) = 0$.

Theorem 6.1 *In the one-variable case, $P =_\sigma Q$ if and only if $\Lambda(P^1 - Q^1) = 0$. Moreover, if $P =_\sigma Q$, then a sequence of valid operations transforming P to Q can be computed in $O(n)$ time.*

Proof: It turns out that if $P =_\sigma Q$, then P can be transformed to Q by using only the equality σ_1. The proof will be constructive and will produce an $O(n)$ time algorithm to give a sequence of transformations from P to Q (or, report that $P \neq_\sigma Q$). The algorithm works in two steps:

Step 1: Transform P to another polynomial P' using the equality σ_1 such that P' differs from P in *at most* one coefficient.

Step 2: Use Corollary 5.1 and transitivity to conclude that $P =_\sigma Q$ if and only if P' is the same as Q.

Since Step 2 is trivial, we concentrate on Step 1. We assume $\Lambda(P) = \Lambda(Q)$ (since otherwise $P \neq_\sigma Q$). We specify how to modify $p_0, p_1, p_2, \ldots, p_{n-2}$, in that order, using valid operations corresponding to the equality σ_1, such that at the end we have $p_0 = q_0$, $p_1 = q_1, \ldots, p_{n-2} = q_{n-2}$. First, since $p_n > 0$, to increase $p_n, p_{n-1}, \ldots, p_0$ by at least 1. This takes $O(n)$ time. Notice that, after this step, $p_i > 0$ for all indices i and the value of $\Lambda(P)$ remains the same as before.

Inductively, assume that before the i^{th} step of coefficient-correction ($i = 0, 1, 2, \ldots, n-2$), we have already corrected the coefficients $p_0, p_1, \ldots, p_{i-1}$ (i.e., have ensured $p_0 = q_0, p_1 = q_1, \ldots, p_{i-1} = q_{i-1}$), possibly by modifying some of the coefficients $p_i, \ldots, p_n$, but **maintaining the following invariant**: $p_i, p_{i+1}, \ldots, p_n > 0$ before, during or after the correction step. We now show how to correct the current value of p_i (i.e., to ensure $p_i = q_i$ if $p_i \neq q_i$) and maintain the invariant that $p_{i+1}, p_{i+2}, \ldots, p_n > 0$. There are three cases to consider:

Case 1. $p_i < q_i$. Then, since $p_{i+1} > 0$, increase both p_i and p_{i+1} by $q_i - p_i$.

Case 2. $p_i > q_i$ and $p_{i+1} > (p_i - q_i)$. Then, decrease both p_i and p_{i+1} by $p_i - q_i$.

Case 3. $p_i > q_i$ and $p_{i+1} \leq (p_i - q_i)$. Then, (since $p_{i+2} > 0$) first increase both p_{i+1} and p_{i+2} by $(p_i - q_i) - p_{i+1} + 1$. Then, decrease both p_i and p_{i+1} by $p_i - q_i$.

Hence, we have changed the coefficients so that all but at most the leading two coefficients differ in the two polynomials. Finally, since $\Lambda(P) = \Lambda(Q)$, and the value of $\Lambda(P)$ does not change by any sequence of valid operations, we must have $p_n - q_n = p_{n-1} - q_{n-1}$. Also, remember that $q_n > 0$ and $q_{n-1} \geq 0$. If $p_n < q_n$, we increase both p_{n-1} and p_n by $p_n - q_n$, otherwise, if $p_n > q_n$, we decrease p_{n-1} and p_n by $p_n - q_n$. It is clear that the total time taken by the above algorithm is $O(n)$. Notice also P can always be transformed to Q provided $\Lambda(P) = \Lambda(Q)$. This completes the proof. ■

7 Closing Comments

Looking for efficient algorithms for finding a label representation is itself a most interesting problem for further research, which is not addressed in this paper. Similarly, the full equivalence problem: "are the *equations* of two given systems the same under some change of variables?" is an obvious next question to tackle. We view this paper as merely a first step in the study of a long list of such questions.

References

[1] R. Alur, T.A. Henzinger, and E.D. Sontag, eds., *Hybrid Systems III. Verification and Control*, Springer Verlag, Berlin, 1996.

[2] L. Blum, M. Shub and S. Smale, "On a theory of computation and complexity over the real numbers", *Bulletin of the American Mathematical Society*, **21**(1989): 1-46.

[3] S. Eilenberg, and M.P. Schüzenberger, "Rational sets in commutative monoids," *J. Algebra* **13**(1969): 173-191.

[4] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness,* W. H. Freeman, 1979.

[5] S. Ginsburg, and E.H. Spanier, "Bounded Algol-like languages," *Trans. Amer. Math. Soc.* **113**(1964): 333-368.

[6] J-P. Quadrat, "Max-plus algebra and applications to system theory and optimal control," in *Proceedings of the International Congress of Mathematicians*, (Zürich, 1994), Birkhäuser, Basel, 1995, pp. 1511–1522,

[7] E. D. Sontag, "Nonlinear regulation: The piecewise linear approach," *IEEE Trans. Autom. Control* **AC-26**(1981): 346-358.

[8] E. D. Sontag, "Remarks on piecewise-linear algebra," *Pacific J.Math.*, **98**(1982): 183-201.

[9] E. D. Sontag, *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, Springer, New York, 1990. (Second Edition, Springer, NY, 1998.)

[10] E. H. Spanier, *Algebraic Topology*, McGraw-Hill, New York, 1966.