Proceedings of the 45th IEEE Conference on Decision & Control
Manchester Grand Hyatt Hotel
San Diego, CA, USA, December 13-15, 2006

Complexity in Automation of SOS Proofs: An Illustrative Example

Dennice Gayme, Maryam Fazel and John C Doyle
Control and Dynamical Systems, California Institute of Technology, Pasadena, CA 91125 USA
Email: *dennice@cds.caltech.edu, maryam@cds.caltech.edu, doyle@cds.caltech.edu*

***Abstract*— We present a case study in proving invariance for a chaotic dynamical system, the logistic map, based on Positivstellensatz refutations, with the aim of studying the problems associated with developing a completely automated proof system. We derive the refutation using two different forms of the Positivstellensatz and compare the results to illustrate the challenges in defining and classifying the 'complexity' of such a proof. The results show the flexibility of the SOS framework in converting a dynamics problem into a semialgebraic one as well as in choosing the form of the proof. Yet it is this very flexibility that complicates the process of automating the proof system and classifying proof 'complexity.'**

I. Introduction

Determining the existence of a solution for a system of polynomial inequalities, non-equalities and equations

$$\begin{aligned} f_i(\mathbf{x}) &\geq 0, \quad i = 1, \ldots, s \\ g_j(\mathbf{x}) &\neq 0, \quad j = 1, \ldots, t \\ h_k(\mathbf{x}) &= 0, \quad k = 1, \ldots, r \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^m$ has emerged as an important field of study in control theory and optimization. This problem has long been of interest in the field of real algebraic geometry where they term the set of $\mathbf{x}$'s that satisfies (1), a (basic) semialgebraic set. In fact, there is an extensive body of computational algebra literature [1] that deals with algorithmic tools for manipulating semialgebraic sets. However, these tools were not widely studied within the control community until relatively recently, in part because of computational intractability. This obstacle was reduced when Parrilo [12], following the pioneering work of Shor [19], introduced a computational framework based on using the existence of a Sum-of-Squares (SOS) decomposition as a sufficient condition for non-negativity of a polynomial. Checking whether a polynomial has an SOS decomposition can be done via semidefinite programming (SDP). This SOS relaxation provides the critical link between the well studied tools from real algebraic geometry and efficient computation.

This framework has been applied to a wide range of problems such as the computation of Lyapunov functions for nonlinear dynamical systems [11], the computation of tight upper bounds for the structured singular value μ [16] in control theory, relaxations for non-convex optimization problems [4] and time delay systems [10]. Related techniques have also been applied to model validation, safety and reachability analysis as introduced in [14]. All of these problems begin as questions about a dynamical system, which are subsequently reduced to algebraic problems (such as proving infeasibility of (1)), using existing ideas from control theory or their extensions. The resulting algebraic question is, in general, NP hard [8], but SOS relaxations and the Positivstellensatz (P-satz) provide a powerful strategy for overcoming this apparent intractability. The Positivstellensatz [20] is a central theorem in real algebraic geometry which basically states that: there is no solution of (1), if and only if there exists a polynomial of a particular form (i.e., satisfying a particular algebraic identity). Such a polynomial is referred to as a P-satz refutation, certificate, or proof.

One of the main advantages of the P-satz is that because it is necessary and sufficient, searching for increasingly higher degree refutations generates a nested hierarchy of polynomial-time computable relaxations (SDPs) which exhausts co-NP (i.e. the proof space). The Matlab toolbox SOSTOOLS, developed by Prajna, Papachristodoulou and Parrilo [15] [16], automates the process of converting an SOS problem to an SDP, which can then be solved by a variety of third party software packages. SOSTOOLS then converts the solution of the SDP into the solution of the original algebraic problem. Typically when using SOSTOOLS we employ Stengle's P-satz [20], less general versions from Schmüdgen [18], Putinar [17], and Handelman [6] can also be used. These forms require more restrictive assumptions, but provide simpler forms for the structure of refutation. Related work by Lasserre [7], based on Putinar's P-satz, focuses on the dual of the SOS problem. The dual of checking the non-negativity of a polynomial over a semialgebraic set is finding a sequence of *moments* that represent a probability measure with support in that set.

This paper has the two main goals of illustrating the application of P-satz methods to solving problems in control and dynamical systems, and presenting the issues associated with further automating SOS proof systems. Analysis of dynamical systems using SOS methods can be broken into three steps. The first is reducing the problem, whether it be feasibility, stability, invariance, safety, reachability, etc. to one involving emptiness of basic semialgebraic sets. The second is attempting to prove emptiness using a P-satz refutation. Finally, one must interpret the output of the proof method. Problem formulation is currently largely handcrafted whereas generating the proof can be automated (albeit with some, often substantial, user expertise required). The step that involves interpretation of the proof output, however, has been the least discussed in the literature. Eventually one would like to automate the whole process but in the present work we focus on issues surrounding the complete

1-4244-0171-2/06/$20.00 ©2006 IEEE.

automation of the last two steps.

The particular application of the P-satz that we focus on is proving robust invariance of sets for the logistic map using SOS methods. The SOS framework also allows us to mirror methods based on considering fixed points and/or periodic orbits and studying the associated attracting sets but here we would like to analyze a larger invariant set. Traditionally for a quadratic recurrence such as the logistic map there are no systematic methods for proving invariance, in part because the system is bifurcating to chaos. Although much is known about the behavior of the logistic map [3] [13], the bulk of this knowledge was gained through simulation or handcrafted proofs which will not scale to higher order systems. Our aim is to not only make definitive statements about a chaotic system, which in this example are not particularly novel, but to do so with methods that potentially scale to higher order systems.

The focus in our discussion of automating proofs is interpreting the output of the proof system. As mentioned above the framework developed by Parrilo et. al. allows one to fix the degree of the polynomials in the P-satz refutation and then check to see that the proper algebraic identity is satisfied. However there is no systematic method for determining the minimum degree polynomials that will generate a certificate. It is often the case that once certificates are obtained it is possible to compare proofs and decide which is 'shortest' in terms of some notion of 'length,' which determines the 'proof complexity.' Unfortunately, all of these quoted terms are currently 'in the eye of the beholder' and there are no rigorous definitions or methods of ordering proof complexity a priori in a way that would allow a systematic search for short proofs. Notions such as 'order of the certificate' and size of the SDP have been proposed but both leave some ambiguity as will be discussed in the sequel. Thus users seem inevitably to be left with some ad hoc choices when searching for short proofs.

The importance of making precise the notion of 'proof complexity' is motivated by the connections it appears to have with important features of the system being analyzed. Of particular interest is the potential to generalize the familiar concept of condition number through the idea that high proof complexity implies problem fragility. There have been complexity bounds established for the Schmüdgen and Putinar versions of the P-satz for basic closed semialgebraic sets consisting solely of inequalities in [21] and [9] respectively, but these require a number of assumptions and are simply bounds on the maximum degree of the polynomials. Other related ideas are discussed in [5]. Here they introduce a P-satz calculus that relies on predicate calculus (inference rules) systematically applied to derive the refutation in a stepwise manner. In this case they defined 'proof complexity' based on the intermediate order of the polynomials and the number of steps. They do not address any method to distinguish between two proofs with the same number of steps and intermediate polynomials of the same order. None of the prior work deals with the problem of evaluating proof complexity for a general system or distinguishing levels of complexity if the order of the refutation is the same.

The logistic map is used as a case study to demonstrate the proof method because it is a familiar system that is known to exhibit interesting behavior. It also allows us to discuss how the selection of the form of the P-satz can change the 'proof order' and to illustrate some of the ambiguities associated with determining a minimum order proof. Further, it is a two dimensional system so one can interpret (visualize) the results graphically. The real value of the tools, of course is for cases when visualization is difficult due to the system dimension. However, to illustrate the ideas and concepts it is useful to start with these simple case studies.

The paper is organized as follows; in section II we provide some preliminary definitions and the P-satz theorem. A brief description of the logistic map follows. Section III presents the invariance proofs and section IV compares the proofs obtained in III to proofs obtained using other forms of the P-satz. Finally conclusions and open problems are discussed.

II. Problem Description

A. Basic Definitions and Theorems

The following three definitions are from [2].

***Definition** 2.1:* Given polynomials $\{g_1, \ldots, g_t\} \in \mathbb{R}[\mathbf{x}]$ the *Multiplicative Monoid* generated by the g_j's, $\mathbf{M}(g_1, \ldots, g_t)$, is the set of all finite products of the g_j's including 1.

***Definition** 2.2:* Given polynomials $\{f_1, \ldots, f_s\} \in \mathbb{R}[\mathbf{x}]$ the *Algebraic Cone* generated by the f_i's is the set

$$\mathbf{C}(f_1, \ldots, f_s) = \left\{ f \;\middle|\; f = \lambda_0 + \sum_i \lambda_i F_i \right\}$$

where $F_i \in \mathbf{M}(f_1, \ldots, f_s)$ and λ_i's are SOS polynomials.

***Definition** 2.3:* Given polynomials $\{h_1, \ldots, h_r\} \in \mathbb{R}[\mathbf{x}]$ the *Ideal* generated by the h_k's is the set

$$\mathbf{I}(h_1, \ldots, h_r) := \left\{ h \;\middle|\; h = \sum_k \mu_k h_k \right\}$$

where $\mu_k \in \mathbb{R}[\mathbf{x}]$.

The solution set of (1) is the basic semialgebraic set

$$\{\mathbf{x} \mid f_i(\mathbf{x}) \geq 0, \quad g_j(\mathbf{x}) \neq 0, \quad h_k(\mathbf{x}) = 0\} \tag{2}$$

for $i = 1, \ldots, s$; $j = 1, \ldots, t$; and $k = 1, \ldots, r$. We call (2) the constraint set and emptiness of this set indicates that (1) is inconsistent. Determining emptiness of a basic semialgebraic set is addressed in the following theorem, due to Stengle [20].

***Theorem** 2.4 (Positivstellensatz [12]):*

$$\{f_i(\mathbf{x}) \geq 0, \quad g_j(\mathbf{x}) \neq 0, \quad h_k(\mathbf{x}) = 0\}$$
$$i = 1, \ldots, s; \; j = 1, \ldots, t; \; k = 1, \ldots, r$$

is infeasible in $\mathbb{R}^n$ if and only if $\exists$ F, G, H such that

$$H + F = -G^2$$
$$F \in \mathbf{C}(f_1, \ldots, f_s)$$
$$G \in \mathbf{M}(g_1, \ldots, g_t)$$
$$H \in \mathbf{I}(h_1, \ldots, h_r)$$

Theorem 2.4 holds for arbitrary systems of polynomial equations, non-equalities and inequalities over the reals. It is remarkable in that it allows one to reduce a geometric problem into an algebraic one without any assumptions on the polynomials. The result is both necessary and sufficient meaning it is always possible to find such a relationship. The main method of generating the infeasibility certificates follows from the fact that by construction the expression $H + F$ is nonnegative and as such it being identically equal to $-G^2$ provides a contradiction. (For further details see [2]).

***Definition** 2.5:* The *subset of the cone* is the set of F_i's from definition 2.2 that are used in the P-satz refutation.

***Definition** 2.6:* The *proof order* is the degree of the highest order term in the P-satz refutation.

***Definition** 2.7:* The *SOS multiplier order* is the order of each of the λ_i's in definition 2.2.

B. System Description

The logistic map is the quadratic recurrence equation

$$x_{k+1} = ax_k(1 - x_k) = Q(x_k) \tag{3}$$

where $a, x_k \in \mathbb{R}$ and $k \in \mathbb{Z}^+$ and each value of the parameter a corresponds to a different dynamical system. It is of interest in both the physics and mathematics communities because it is a simple iterative equation that is known to exhibit chaotic behavior [3]. The most common use of (3) is as model of population dynamics [13] in Ecology but it can also be used to model other phenomena. In figure 1 the points indicate the

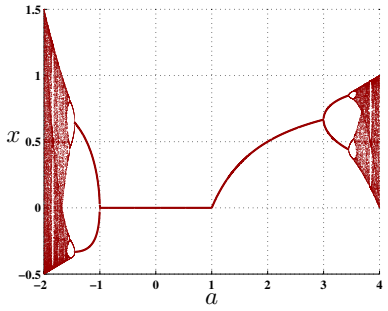

Fig. 1. Logistic Map Attractors $-2 < a < 4$

attracting set or steady state value of (3) for each parameter a. The map has two fixed points; $x = 0$ is the attracting fixed point for $-1 < a < 1$. At $a = 1$ there is a transcritical bifurcation and $x = 1 - \frac{1}{a}$ becomes the attracting fixed point until $a = 3$ when there is a period doubling bifurcation. There is a similar critical point and bifurcation at $a = -1$. The two period attracting sets continue until there is another set of period doubling bifurcations at $1 + \sqrt{6}$ and $1 - \sqrt{6}$ respectively. The system then continues to undergo period doubling bifurcations to chaos, as shown in the right and left portions of figure 1.

III. Invariance Proofs

We focus on invariance of the largest closed invariant set. Note that the complement of that set is also invariant as all points in the complement eventually become unbounded. The proof method discussed herein allows one to search for refutations that guarantee emptiness of basic semialgebraic sets. In general, the union of basic semialgebraic sets does not necessarily yield a basic semialgebraic set, so in some cases we will not be able to do a proof search over the union directly. However, this is not a limitation of the method because any semialgebraic set can be written as the union of basic semialgebraic sets [2]. This guarantees that we can always break a given semialgebraic set into subsets and since the union of two empty sets is clearly empty one can make a determination about the entire region of interest by using separate proofs for each subset. This approach is reminiscent of branch-and-bound methods used in global optimization, where the feasible set is divided iteratively by branching on the decision variables. Although we will not pursue it in this paper, a similar approach can be used for proving a set to be empty, we refer to this approach as 'SOS branch-and-prove'. In this example we begin with two branches;

$$\{(2x-1)^2 - 1 \leq 0; (a-1)(a-4) \leq 0\} \tag{4}$$

$$\{a^2(2x-1)^2 - (a-2)^2 \leq 0; (a+2)(a-1) \leq 0\} \tag{5}$$

based on the geometry of the region of interest, which is

$$\begin{array}{rr} 0 \leq x \leq 1 & 1 \leq a \leq 4 \\ 1 - \frac{1}{a} \leq x \leq \frac{1}{a} & 0 \leq a \leq 1 \\ \frac{1}{a} \leq x \leq 1 - \frac{1}{a} & -2 \leq a \leq 0. \end{array} \tag{6}$$

Figure 2 outlines the region (6) in heavy black lines.

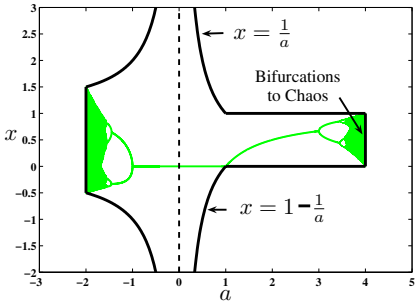

Fig. 2. Invariant Set: Logistic Map

A. Branch 2: $1 \leq a \leq 4$

For the region described by the set (4) one can verify invariance through the following proposition;

***Proposition** 3.1:* Given $a > 0$ the region $0 \leq x \leq 1$ is invariant if and only if $0 < a \leq 4$.

Proof:

Sufficiency Given $a > 0$ and $\{x \mid 0 \leq x \leq 1\}$ is invariant,
$\Rightarrow\ 0 \leq ax(1-x) \leq 1\ \Rightarrow\ 0 \leq x(1-x) \leq \frac{1}{a}$.
Note $x(1-x)$ is a parabola with a max at $x = \frac{1}{2}$

$$\Rightarrow \frac{1}{4} \leq \frac{1}{a}\ \Rightarrow\ a \leq 4.$$

Necessity Given $0 < a \leq 4$ and $0 \leq x \leq 1$.
$\Rightarrow 0 \leq x(1-x) \leq \frac{1}{4}\ \Rightarrow\ 0 \leq ax(1-x) \leq a\frac{1}{4} \leq 1$ ■

This proof is hard to automate because it relies on observations about the system and heuristics. It also requires knowledge of regions outside the set of interest. Such a proof method is system dependent and lacks the flexibility to analyze different aspects of the system with the same steps.

For a more general proof method one can appeal to the machinery provided in theorem 2.4. Here proving invariance is equivalent to proving

$$\left\{\begin{array}{r}(2x-1)^2 \leq 1\\ (a-1)(a-4) \leq 0\\ (2ax(1-x)-1)^2 > 1\end{array}\right\} = \varnothing \tag{7}$$

Condition (7) means that for all x such that $(2x-1)^2 \leq 1$ future points in the orbit of the logistic map, $Q(x)$, will never intersect the region outside of $(2x-1)^2 \leq 1$, (i.e. all future points remain inside).

In this framework a constraint set for (7) is defined by

$$\begin{aligned} f_1(a,x) &= (2x-1)^2 - 1 \leq 0\\ f_2(a,x) &= (a-1)(a-4) \leq 0\\ f_3(a,x) &= 1-(2ax(1-x)-1)^2 \leq 0\\ f_3(a,x) &\neq 0. \end{aligned} \tag{8}$$

Given this constraint set the P-satz refutation is of the form

$$0 = -f_3^2 - p_{13}f_1f_3 + p_{123}f_1f_2f_3, \quad \text{where} \tag{9}$$

$$\begin{aligned} p_{13}(a,x) &= \frac{4}{3} - \frac{2}{3}a + \frac{1}{3}a^2 - xa^2 + x^2a^2\\ p_{123}(a,x) &= \frac{1}{3}. \end{aligned} \tag{10}$$

Noting that $f_3 = -a(ax^2 - xa + 1)f_1$ and $p_{13} = \frac{1}{3}f_2 + a(x^2a - ax + 1)$, (9) can be written as

$$-f_3^2 - \frac{1}{3}f_2f_1f_3 + f_3^2 + \frac{1}{3}f_1f_2f_3 = 0. \tag{11}$$

B. Branch 2: $-2 \leq a \leq 1$

The proof in (9) gives us both necessary and sufficient conditions for invariance of the region (4) using a more general proof method that extends to higher order systems and alternative definitions of f_1, f_2 and f_3. For example one can manipulate (5) into

$$\begin{aligned} f_1(a,x) &= a^2(2x-1)^2 - (a-2)^2 \leq 0\\ f_2(a,x) &= (a+2)(a-1) \leq 0\\ f_3(a,x) &= (a-2)^2 - a^2(2ax(1-x)-1)^2 \leq 0\\ f_3(a,x) &\neq 0 \end{aligned}$$

and employ (9), exactly the same P-satz refutation with

$$\begin{aligned} p_{13}(a,x) &= \frac{1}{3} + \frac{1}{3}a + \frac{1}{3}a^2 - xa^2 + x^2a^2\\ p_{123}(a,x) &= \frac{1}{3}. \end{aligned}$$

Substituting $p_{13} = \frac{1}{3}f_2 + (a^2x^2 - xa^2 + 1)$ and $f_3 = -f_1(a^2x^2 - xa^2 + 1)$ into (9)

$$-f_3^2 - \frac{1}{3}f_2f_1f_3 + f_3^2 + \frac{1}{3}f_1f_2f_3 = 0.$$

It is clear that the proof of proposition 3.1 could not be directly applied for the branch $-2 \leq a \leq 1$, and that the SOS method is far more general.

IV. Proof Complexity and Automation of Proofs

The following discussion focuses on the branch with $1 \leq a \leq 4$, but the basic ideas generalize. In section III the proofs were generated using the Stengle version of the P-satz (Theorem 2.4). This theorem implies that if we search over the entire cone, monoid and ideal of the inequalities, non-equalities and equalities respectively, and find some F, G and H that satisfy the identity, then the set is empty. This is very general and in practice we choose to start with some subset of each of these objects. For example given a system such as (8), one tends to begin by setting the multiplicative monoid of the non-equality equal to $\{f_3\}$. Then we select some subset of $\mathbf{C}(f_1, f_2, f_3)$ such as the first three terms of the general expression

$$p_0 + \sum_i p_i f_i + \sum_{\{i,j\}} p_{ij} f_i f_j + \sum_{\{i,j,k\}} p_{ijk} f_i f_j f_k + \dots \tag{12}$$

from definition 2.2. Initially the order of the SOS multipliers (p_i's) is selected to ensure that each expression in the refutation has the same degree (usually in each variable). If this does not yield a certificate one can proceed in a number of ways. These include: adding or subtracting terms from the subset of the cone (and/or multiplicative monoid), or increasing the order of the certificate or a combination of both. There are also a number of heuristics that can be used depending on what caused the refutation to fail. If for example SOSTOOLS gives a positive answer with numerical errors, removing the monomials in the SOS multipliers that have 'very small' coefficients can often resolve the issue. Experience using SOSTOOLS and SDP solvers allows a user to develop additional intuition regarding how to change things for a specific problem.

Clearly, this procedure leaves a great deal of flexibility in forming the refutation, both a strength and a weakness. In the present work this flexibility was exploited in order to generate proofs that were easy to write down and check. There was a structure that allowed one of the f_i's to be written in terms of another, which made using a combination of pairwise products and the triple product of the inequalities to form the cone a natural choice. Intuition allowed us to manipulate the form of the proof by hand, but currently this cannot be automated.

Alternatively, one may want to come up with a proof using only the first two terms of (12). In fact, Putinar in [17] showed that given a compact set

$$K = \{\mathbf{x} \mid q_i(\mathbf{x}) \geq 0,\ i = 1, \dots, n\}$$

where the highest degree homogeneous parts of the q_i's have no common zeros in $\mathbb{R}^n$, except at 0, then any strictly positive q_0 on K belongs to the additive cone $p_0 + \sum_{i=1}^n p_i q_i$.

This theorem implies that for the branch $1 \leq a \leq 4$, if $f_3 + p_1f_1 + p_2f_2 = p_0$, where p_i's are SOS polynomials then

$$\{f_1 \leq 0, f_2 \leq 0,\ f_3 \leq 0,\ f_3 \neq 0\} = \varnothing. \tag{13}$$

If we apply this form we get a refutation where the order of the SOS multipliers are shown in table I. This refutation is

Polynomial	Order in x	Order in a
p_0	8	6
p_1	6	6
p_2	8	4

TABLE I

higher order than the refutation (9) and so it is easy to see that the proof is longer. It turns out that if we instead apply Putinar's P-satz to the equivalent set

$$\left\{f_1 \leq 0, f_2 \leq 0,\ f_3 \leq 0,\ f_3^2 > 0\right\} \tag{14}$$

the refutation becomes

$$-f_3^2 + p_1 f_1 + p_2 f_2 + p_3 f_3 = p_0. \tag{15}$$

The order of each of the SOS multipliers is shown in table II. This refutation (15) only uses the SOS multipliers of the first order monomials in the cone, so one may argue it is a simpler form than (9). In fact, the Putinar P-satz may be

Polynomial	Order in x	Order in a
p_0	8	4
p_1	6	4
p_2	8	2
p_3	4	2

TABLE II

preferable when automating a proof system for low order systems because it is known that a proof of this form always exists [17], (under the assumption mentioned above). This allows one to iteratively search for a proof by changing only the degree of the SOS multipliers. However, the resulting proof may not be computationally tractable due to the size of the associated SDP.

The order of the refutations (9) and (15) are the same, therefore if we were to define 'proof complexity' as simply the order of the proof both of these proofs would have the same 'length'. It is clear however that the proof (9) is much easier to verify and makes a lot more sense to a human, than SOS multipliers that contain every monomial with degree of $x \leq 6$ and $a \leq 4$ as is the case for p_1 in (15). In this case it seems that the order of the SOS multipliers may be a better measure, but this is not true in all cases. In larger systems all of the SOS multipliers may be a very high order with many monomials, and then the method of comparison becomes more challenging.

An appealing alternate means of classifying 'proof complexity' is the computational cost of solving the corresponding SDP, which is a function of the size and conditioning of the SDP. The discussion in the sequel is based on the SeDuMi solver (for details see [22]). The SDP size is given by the dimension of A in

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & A^T x = b, \quad X \in \mathcal{K} \end{array} \tag{16}$$

where $x = \mathbf{vec}(X)$, and $\mathcal{K}$ denotes the cone of positive semidefinite matrices. The conditioning is related to the relative magnitude of the residual $\|A^T x - b\|$ which is returned by the SDP solver. For the details of transforming an SOS problem into (16) see [15].

Searching for a valid certificate of the form (9), where the degree of p_{13} and p_{123} are selected so that the degree in a and x is the same for every term, requires solving an SDP that has size 66×25. Similarly finding a valid certificate using the form (15) where the SOS multipliers are again chosen to match the degree of each term in the equation, (as in table II), requires solving an SDP with size 485×45. Given this one may be tempted to conclude that the refutation (9) is 'shorter' or has lower 'complexity' than (15), which certainly seems to better agree with human perception. However there are some ambiguities associated with these numbers (for example a problem with a sparse A matrix is likely much easier to solve than a dense one), and conditioning has not been taken into account.

In fact, these numbers are merely representative for proofs that have the same basic structure as those discussed herein because it is hard to quantify the hand manipulations and heuristics that went into generating (9) and (15). For example in (9) we set $-f_3^2 - p_{13} f_1 f_3 + p_{123} f_1 f_2 f_3 = p_0$ and forced the coefficients of p_0 to be small. Some of the monomials in both p_{13} and p_{123} were also removed because their coefficients were very small compared to the others. Making these changes required some additional constraints which increased the size of the SDP to 283×74. This new number is not really an accurate characterization of 'proof size' because implementation of these extra constraints is user dependent and a user with more expertise may have been able to achieve the same goals with a lesser increase in computational cost. Further the size that results from heuristic changes may be an artifact of the way the SOSTOOLS software was designed/implemented, (i.e. some heuristics may make the software do things in a less than optimal manner). In order to make the comparison one could make the same changes to the proof in (15) and then try to compare SDP sizes. However, attempting to force p_0 in (15) to be equal to zero results in there being no refutation of the same order. Also, manually manipulating terms is much harder and may require a larger addition of constraints because the p_i's are higher order. It is also possible that different heuristics can be applied to improve certain properties of (15) and these may change the size of that SDP.

For all of the results shown in this paper the SDP is ill-conditioned, which is not surprising because the question being asked is not robust. If one were to shift the border of the region in figure 2 outwards by any amount the region would no longer be invariant. For the logistic map the equation describing the invariant set is simple but in many cases the boundary of the invariant region of a chaotic system is fractal. For example if one were to look at the complex version ($x, a \in \mathbb{C}$) of the same map, (which would effectively double the order of the problem) one would get a parameterization of the Mandelbrot set. The problem of identifying the boundary of the Mandelbrot set is known to be undecidable in the sense of Turing.

V. Conclusions

In this paper we presented a case study using a method for proving invariance in a chaotic dynamical system based on P-satz refutations. Construction of the proofs was carried out using Stengle's P-satz with the goal of minimizing the order of the SOS multipliers and the total number of terms in the refutation. The flexibility of the proof system was illustrated by generating two additional proofs using Putinar's form of P-satz. These additional proofs also allowed us to demonstrate the critical nature of problem definition since defining the set (13) slightly differently as in (14) allowed us to generate a lower order proof. The order of the SOS multipliers for the original proofs was then compared to the order of SOS multipliers obtained using Putinar's form of P-satz with the same order certificate. The accompanying discussion illustrated the current difficulty with trying to define a minimum order proof.

SOS methods are very powerful but their use has not yet reached an ideal level of automation. They have a lot of flexibility which in many cases is beneficial but can also be problematic. The advantage is that a tremendous variety of problems can be reduced to something that SOS methods, (particularly through SOSTOOLS) can address. This makes it the most promising research direction to pursue for a remarkably broad class of problems in controls, dynamical systems, and many other areas. The problem arises because there are many ways to formulate the constraint set for a given problem, and many ways to structure the refutation and we do not yet have a means to determine optimal choices. Further, once a certificate has been obtained, a systematic method of comparing or classifying proof complexity even for proofs of the same order is not well understood.

VI. Open Problems

In this paper we focused on proving particular properties of a dynamical system after the problem had been reduced to the form (1). Yet as was shown in section IV the ability of SOSTOOLS to find a low order refutation is closely tied to how the problem is formulated. Another example of this issue was in section III where the choice to express the constraint $0 \leq x \leq 1$ as $(2x-1)^2 - 1$ was not arbitrary, in fact the problem formulation was iterated along with the search for a refutation. This iteration puts a human into the loop and a systematic means of finding an optimal problem formulation is still lacking. In this case the handcrafted method is fine because the problem is low order and has few constraints but this aspect of the method does not always scale well.

Even with a systematic method to formulate problems as semialgebraic sets further research is needed in the implementation of the so called 'SOS branch and prove' method. In the present work the invariant set was the union of two basic semialgebraic sets. In general, especially in problems of higher order this will not be the case and deciding when to branch and finding appropriate basic semialgebraic subsets, so far, is problem dependent and not general.

Some parts of constructing the proof (finding a refutation) require a human in the loop, even when using SOSTOOLS. If one does not obtain a proof using the initial subset of the cone, monoid or ideal the path to fix the problem is not unique. Approaches such as manually checking the magnitude of each coefficient of each SOS multipliers and then eliminating the ones that are small compared to others can be automated more easily than intuition. Intuition helps a user choose the terms to include and to manipulate the degree of the terms in the refutation, however it relies on clever tricks that do not scale well.

VII. Acknowledgments

The authors would like to thank Stephen Prajna for his great patience, many helpful suggestions and discussions.

References

[1] S. Basu, R. Pollack, and M.F. Roy. *Algorithms in Real Algebraic Geometry*. Springer-Verlag, 2003.

[2] J. Bochnak, M. Coste, and M. F. Roy. *Real Algebraic Geometry*. A Series of Modern Surveys in Mathematics. Springer-Verlag, 1998.

[3] R. Devaney. *An Introduction to Chaotic Dynamical Systems*. Addison-Wesley, 2nd edition, 1989.

[4] M. Fazel and M. Chiang. Network utility maximization with nonconcave utilities using Sum-of-Squares method. In *Proceedings of the 44th IEEE Control and Decision Conference*, 2005.

[5] D. Grigoriev and N. Vorobjov. Complexity of Null- and Positivstellensatz proofs. *Annals Pure and Applied Logic*, 113:153–160, 2001.

[6] D. Handelman. Representing polynomials by positive linear functions on compact convex polyhedra. *Pacific J. Math.*, 132:35–62, 1988.

[7] J. B. Lasserre. Global optimization with polynomials and the problems of moments. *SIAM Journal of Optimization*, 11:796–817, 2001.

[8] K. G. Murty and S. N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Math. Prog.*, 39:117–129, 1987.

[9] J. Nie and M. Schweighofer. On the complexity of Putinar's Positivstellensatz. *eprint arXiv:math/0510309*, October 2005.

[10] A. Papachristodoulou. *Scalable Analysis of Nonlinear Systems Using Convex Optimization*. PhD thesis, Caltech, Pasadena, CA, 2005.

[11] A. Papachristodoulou and S. Prajna. On the construction of Lyapunov functions using the Sum of Squares decomposition. In *Proceedings of the 41st IEEE Conf. on Decision and Control*, 2002.

[12] P. A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, Caltech, Pasadena, CA, 2001.

[13] H. O. Peitgen, H Jurgens, and D. Saupe. *Chaos and Fractals: New Frontiers of Science*. Springer-Verlag, 1992.

[14] S. Prajna. *Optimization-Based Methods for Nonlinear and Hybrid Systems Verification*. PhD thesis, Caltech, Pasadena, CA, 2005.

[15] S. Prajna, A. Papachristodoulou, and P. A. Parrilo. *SOSTOOLS: Sum of Squares Optimization Toolbox for MATLAB: User's guide*. Caltech, Pasadena, California, 2002. Available at *http://www.cds.caltech.edu/sostools*.

[16] S. Prajna, A. Papachristodoulou, and P.A. Parrilo. Introducing SOSTOOLS: A general purpose Sum of Squares programming solver. In *Proceedings of the 41st IEEE Conf. on Decision and Control*, 2002.

[17] M. Putinar. Positive polynomials on compact semi-algebraic sets. *Indiana Univ. Math. J.*, 42:969–984, 1993.

[18] K. Schmudgen. The k-moment problem for compact semialgebraic sets. *Math. Ann.*, 289:203–206, 1991.

[19] N. Z. Shor. Class of global minimum bounds of polynomial functions. *Cybernetics and Systems Analysis*, 23:731–734, 1987.

[20] G. Stengle. A Nullstellensatz and a Positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207:67–97, 1974.

[21] G. Stengle. Complexity estimates for the Schmudgen Positivstellensatz. *J. Complexity*, 12(2):167–174, 1996.

[22] J. F. Strum. Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones (updated for version 1.05). *Optimization Methods and Software*, 11–12:625–653, 1999 (for verions 1.02/1.03). Updated in 2001 and available from *http://sedumi.mcmaster.ca*.