# Abstracting Continuous Nonpolynomial Dynamical Systems

**William Denman**

**Computer Laboratory, University of Cambridge, UK**
william.denman@cl.cam.ac.uk

**Abstract**

The reachability problem, whether some unsafe state can be reached, is known to be undecidable for nonlinear dynamical systems. However, finite-state abstractions have successfully been used for safety verification. This paper presents a method for automatically abstracting nonpolynomial systems that do not have analytical or closed form solutions.

The abstraction is constructed by splitting up the state-space using nonpolynomial Lyapunov functions. These functions place guarantees on the behaviour of the system without requiring the explicit calculation of trajectories. MetiTarski, an automated theorem prover for special functions (sin, cos, sqrt, exp) is used to identify possible transitions between the abstract states. The resulting finite-state system is perfectly suited for verification by a model checker.

**1998 ACM Subject Classification** I.6.4 Model Validation and Analysis

**Keywords and phrases** Formal Verification, Automated Theorem Proving, Abstraction, Non-polynomial System, MetiTarski

**Digital Object Identifier** 10.4230/OASIcs.ICCSW.2012.42

## 1 Introduction

Abstracting continuous systems into a finite state representation has been a successful method for the formal verification of real world problems. Current abstraction methods can only handle linear or polynomial nonlinear systems. Nonpolynomial terms must be either linearised or over-approximated [4]. These approximations can introduce abstract states that are seen as false-positives by the model checker. In this paper, the automated theorem prover MetiTarski is used to create an abstraction of a nonpolynomial continuous system by working with the nonpolynomial terms directly. This goal is to enhance the quality of the resulting finite state abstraction.

MetiTarski [2] is an automated theorem prover for arithmetical conjectures involving transcendental functions (sin, cos, exp etc.). It has been successful in proving arithmetical theorems that are used to verify analogue circuits [3] and linear hybrid systems [1].

Most systems of interest can only be specified using nonlinear differential equations. This is because, not surprisingly, nonlinear systems present a richer set of dynamics. It is for these reasons that both qualitative analysis and repeated numerical simulation is used [10]. The finite level of precision of numerical methods is often a source of significant error.

Safety, the fact that some bad behaviour will never happen, is the most important property that should be verified for a system. The reachability computation remains the most common way to check safety of a system. Unfortunately, the reachability decision problem of continuous systems is undecidable [6]. Abstraction methods are commonly employed to solve this problem.

By abstracting properly and preserving the relevant underlying behaviour of the system, tools that are already developed can be used. Sloth and Wisniewski [12] developed a method for creating a sound and complete abstraction of continuous systems using Lyapunov functions. By using the Lyapunov function as a predicate for partitioning, they were able to convert the infinite state space of a continuous system into timed automata. They are however only able to abstract a restricted class of linear systems.

Another abstraction method borrows ideas from the domain of qualitative reasoning. Qualitative reasoning is motivated by the idea that numerical simulation is limited when not all the parameters of the system are known. Instead of trying to compute a solution, it is sufficient to look at how the vector field itself changes over time. Tiwari [14] uses predicates that evaluate over the three symbols $\{+, -, 0\}$ to split up the infinite state space. This construction of the abstraction uses the decidability of the first order theory of real closed fields [13] to compute the transitions between abstract states. Once the abstraction is created then a model checker is used to evaluate Computation Tree Logic (CTL) properties on the abstract system. The method proposed by Tiwari is limited to nonlinear polynomial vector fields.

## 2 Dynamical Systems

A dynamical system can be thought of as an abstract entity that changes its behaviour and state with respect to time. The *state* is the current value of the variables of the system. The *behaviour* is a function that returns the next state of the system, given the current state. These two quantities are required to completely model the system.

▶ **Definition 1** (Dynamical System). An $n$-dimensional dynamical system $DS$ is represented by the state vector $\mathbf{x}(t) \in \mathcal{R}^n$ and a function $f : \mathcal{R}^n \to \mathcal{R}^n$

For continuous systems time will progress as a smooth function. Instead of giving an explicit value of the next state, function $f$ will define how the system evolves continuously. The simplest way to model this smooth change of variables is using *differential* equations.

▶ **Definition 2** (Continuous Dynamical System). A continuous dynamical system is compactly modelled using a set of differential equations of the form

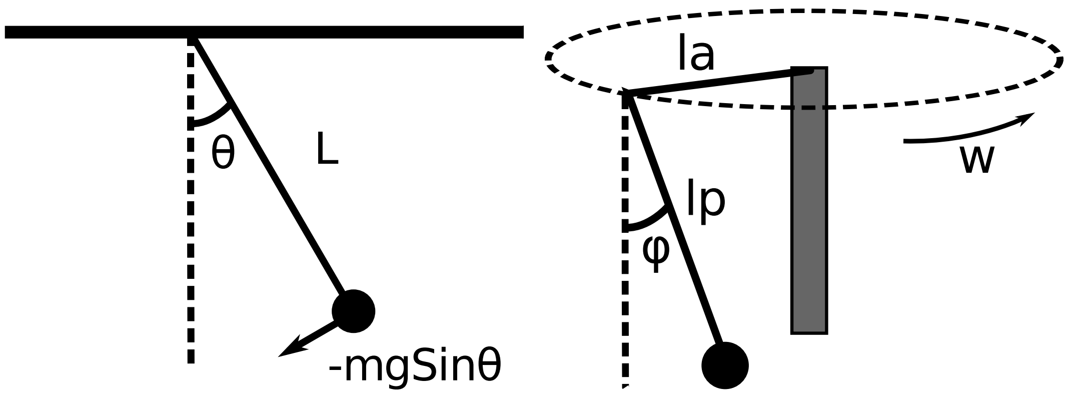
$$\mathbf{x}'(t) = f(\mathbf{x}(t)) \tag{1}$$

It is common convention to drop the explicit reference to the time variable.

$$\mathbf{x}' = f(\mathbf{x}) \tag{2}$$

The benefit of using differential equations is that continuous systems can be completely represented by how their variables change. Even for the simplest of systems, it can be quite difficult and in most cases impossible to analytically solve Equation (1). If the functions $f(x)$ are polynomial, that is $f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$ where n is a non-negative integer and $a_0, a_, .., a_n$ are constants then the resulting system is polynomial. Otherwise the system is nonpolynomial.

▶ **Example 1** (The Pendulum). Take for instance the friction-free pendulum of Figure 1a. A rod of length $L$ is attached to a ball of mass $m$. As the ball swings, the angle $\theta$ between the rod and the vertical changes. The angular velocity (rotational speed in the tangential direction) $\omega(t)$ is equivalent to the change of the angle $\theta$ or $\frac{d\theta}{dt}$. Acceleration, velocity and position of the ball are related by $a = v' = x''$. The arc-distance travelled by the ball is $x = \theta L$. The effective force returning the ball to the center is $mg \sin\theta$. The differential



**(a)** Regular pendulum **(b)** Whirling pendulum

**Figure 1** Two nonpolynomial systems.

equations of the system can be derived from Newton's $2^{\text{nd}}$ Law $F = ma$. Taking $\frac{F}{m} = a$, $a = x'' = (\theta L)'' = \omega' L$ gives the system in state space form

$$\theta' = \omega \tag{3a}$$

$$\omega' = -\frac{g}{L}\sin\theta \tag{3b}$$

This model is exactly described by two simple differential equations. The problem is that there is no known method that can obtain a solution with respect to time for either of the state variables ($\theta(t)$ or $\omega(t)$). This is due to the nonpolynomial $\sin\theta$ term in Equation (3b). Under certain conditions, nonpolynomial systems like Example 1 can be approximated by a *linear* system that is guaranteed to have a closed form solution.

▶ **Definition 3** (Linear System). When $f(\mathbf{x})$ (Equation (2)) is defined by an affine line $ax + b$, the continuous system is said to be linear. If the state vector $\mathbf{x}$ is $n$-dimensional then, $f(\mathbf{x}) = Ax + b$ where A is an n by n matrix b is an n vector.

Since the linear system is defined using a square matrix, it is easy to extract the eigenvalues [11]. These eigenvalues can be used to construct the solution to the system of equations and to understand the qualitative behaviour of the system's trajectories. The interesting result is that if a nonpolynomial system is replaced by a linear approximation, the eigenvalues of the approximation can be used to understand the behaviour of the original system. The linear approximation is only valid in a close neighbourhood of a particular point. In dynamical system analysis, this is usually chosen to be an *equilibrium point.*

▶ **Definition 4** (Equilibrium Point). An equilibrium or fixed point is a location in the state-space $\tilde{\mathbf{x}}$ where $f(\tilde{\mathbf{x}}) = \tilde{\mathbf{x}}$. When the system is at the equilibrium point, it will stay there for all time if not disturbed. If a slight disturbance causes the system to leave the equilibrium point and never return then the equilibrium point is *unstable*, otherwise the system is *stable*.

Since nonpolynomial systems cannot be solved analytically, verification relies on the analysis of the qualitative behaviour of the system near its equilibrium points. This *qualitative analysis* looks to see how sets of trajectories move in the state-space. For linear systems, if the eigenvalues of the system all have a negative real part, then the stability of the equilibrium point is guaranteed. If the real parts of the eigenvalues are all 0 then nothing can be concluded and the linearisation method fails. An alternative method that operates on the original nonlinear vector field directly can be used instead.

▶ **Definition 5** (Lyapunov Function). A function $V(x)$ is a Lyapunov function, if for an equilibrium point (fixpoint) located at the origin (0,0) the following conditions hold

$$V(x) > 0 \qquad \text{for } x \neq 0 \tag{4a}$$

$$V(0) = 0 \tag{4b}$$

$$\frac{\partial V(x)}{\partial t} \leq 0 \qquad \text{for all } x \tag{4c}$$

If a Lyapunov function exists, then the equilibrium point is guaranteed to be stable [10]. The Lyapunov property is a sufficient condition for stability.

Return to Example 1 but assume now that the system is real by including friction effects. $V(x)$ can be chosen as the total energy (kinetic plus potential) of the system. It is clear that when the pendulum is displaced, energy is put into the system causing $V(x)$ to increase and $V(x) > 0$ . The energy of the system will only be zero when the pendulum has stopped swinging and is hanging straight down at position 0, therefore $V(0) = 0$. The system continuously loses energy due to friction and $V(x)$ is always decreasing, implying that $V'(x) < 0$. Since the three constraints have been met, $V(x)$ is an Lyapunov function and by definition the equilibrium point at rest is stable.

The Lyapunov method does not require that $V(x)$ be the energy of the system, any function can be used. The caveat is that finding a Lyapunov function in general can be quite difficult. There are several advanced methods based on sum-of-squares (SOS) techniques that make the search for the Lyapunov function tractable. These methods have been implemented in a MATLAB package called SOSTOOLS [9]. The next section describes an abstraction algorithm that uses Lyapunov functions.

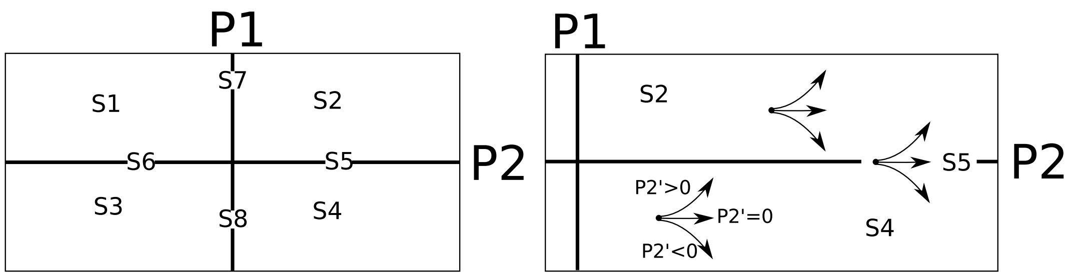
## 3 Abstracting the Dynamical System

The end-goal of verification is to prove that a system has been built correctly. For continuous systems we specifically want to prove that all trajectories starting in a *safe* state will never reach a *bad* or *unsafe* state. This reachability analysis is known to be decidable for discrete systems such as finite automata, but it is undecidable for nonpolynomial systems. Abstraction methods are a shortcut used to obtain a decidability result from the undecidable brick wall.

The abstraction method of Tiwari [14] discretizes the state-space using predicates evaluated over three symbols $\{+, -, 0\}$. Each abstract state is defined by a conjunction of these predicates. For the example in Figure 2a, predicates P1 and P2 represented by thick lines have been used to discretize a two dimensional state space. Taking P1 : $f(x, y) = x$ and P2 : $f(x, y) = y$, state S1 is $P1 > 0 \wedge P2 > 0$, S7 is $P1 = 0 \wedge P2 > 0$ and so on.

The difficulty in creating finite state abstractions is choosing the predicates used to discretize the state space. Tiwari has developed several heuristics for defining *good* predicates. Lyapunov functions are a good choice because they represent a positively invariant set. By definition, the solutions of the system will only pass through the level sets of Lyapunov functions in one direction. Including Lyapunov functions greatly simplifies the construction of abstract transition relations by limiting the reachable state space.

A decision procedure is used to determine all possible transitions between the abstract states. For instance, in the example shown in Figure 2a the following cases must be checked to determine the transitions between S2, S4 and S5. For brevity, P1 is assumed to be positive. Transitions between abstract states are decided by checking the sign of the derivative of the predicate with respect to the vector field of the system. To take this derivative, the chain rule for partial derivatives is used $\frac{dPn}{dt} = \frac{\partial Pn}{\partial x}\frac{dx}{dt} + \frac{\partial Pn}{\partial y}\frac{dy}{dt}$.



**(a)** Discretizing the state space with predicates **(b)** Transitioning between abstract states

**Figure 2** Tiwari's Abstraction Method.

1. If the current state is S4 : (P2 $<$ 0) then:
   - If $P2' > 0$, the next state is either S4 or S5
   - If $P2' = 0$, the next state is S4
   - If $P2' < 0$, the next state is S4
   - If unknown, the next state is S4 or S5
2. If the current state is S5 : (P2 $=$ 0) then:
   - If $P2' > 0$, the next state is S2
   - If $P2' = 0$, the next state is S5
   - If $P2' < 0$, the next state is S4
   - If unknown, the next state is S2 or S5 or S4
3. If the current state is S2 : (P2 $>$ 0) then:
   - If $P2' > 0$, the next state is S2
   - If $P2' = 0$, the next state is S2
   - If $P2' < 0$, the next state is S5 or S2
   - If unknown, the next state is S2 or S5

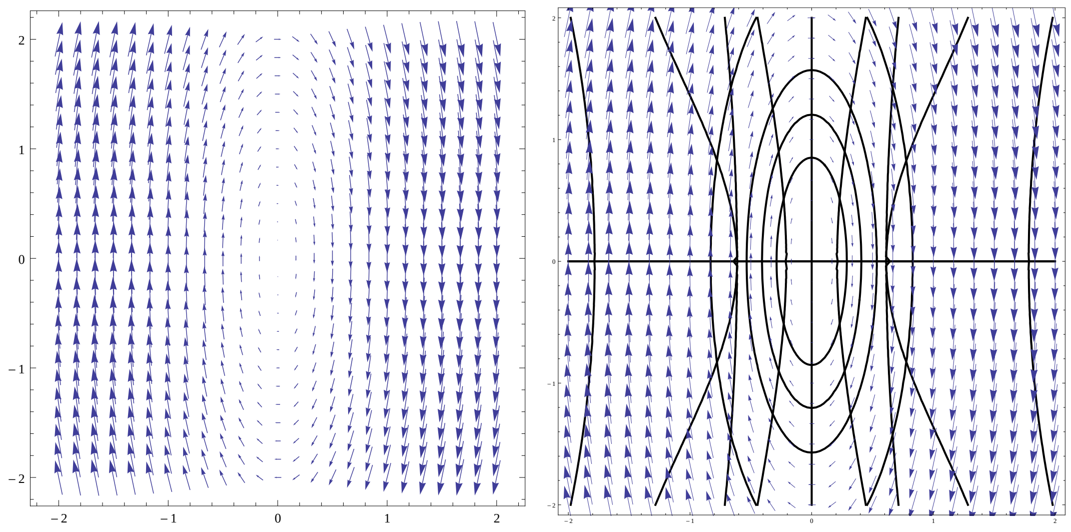
One issue is that the decision procedure used by Tiwari is only applicable to polynomials. This restriction limits the type of systems that can be analysed. The next example shows how Tiwari's method is extended to work with nonpolynomial systems. MetiTarski is used to reason about the inequalities that are generated during the abstract transition analysis described above.

▶ **Example 2** (Whirling Pendulum). Consider Figure 1b, where a pendulum of length $l_p$ is attached to a movable rigid arm of length $l_a$. Taking the following assumptions: the pendulum is light enough to be swung up with a small $\phi'$, ignore friction and consider that each pendulum arm is thin enough to make the moment of inertia negligible [5]. With $x_1 = \phi$ and $x_2 = \phi'$ the system of equations are

$$x_1' = x_2 \tag{5a}$$

$$x_2' = \omega^2 \sin x_1 \cos x_1 - \frac{g}{l_p} \sin x_1 \tag{5b}$$

The predicates used to split up the state space are obtained by repeatedly taking the



**(a)** Vector field **(b)** Discretized state-space

**Figure 3** The whirling pendulum system.

derivative of the vector field.

$$P1 = x_2'' = -x_2 \sin^2(x_1) + x_2 \cos^2(x_1) - 10x_2 \cos(x_1) \quad (6a)$$

$$\begin{aligned} P2 = P1' = \frac{1}{4} \sin(x_1)(8{x_2}^4(8\cos(x_1) - 5) - 8{x_2}^2(561\cos(x_1) - 210\cos(2x_1) \\ + 11\cos(3x_1) - 65) - 1104\cos(x_1) - 8040\cos(2x_1) + 2304\cos(3x_1) \\ - 175\cos(4x_1) + 4\cos(5x_1) + 4095) \quad (6b) \end{aligned}$$

and a Lyapunov function of the system is found using SOSTOOLS

$$V1 = 0.3345x_2^2 + 1.4615\sin^2 x_1 + 1.7959\cos^2 x_2 - 6.689x_2 + 4.8931 \quad (7)$$

The behaviour of the system is shown in the vector field plot of Figure 3a with $x_1$ on the horizontal axis and $x_2$ on the vertical axis. Using the predicates obtained from the equations of the system (Equations 6a and 6b) along with several level sets of the Lyapunov function (Equation $7:\ V1 = 0.25, V1 = 0.5, V1 = 0.85$ and $V1 = 2$), the state-space is discretized as shown in Figure 3b. MetiTarski is used to determine the transitions between the abstract states using the method described in Section 3. The methods of Sloth, Wisniewski and Tiwari cannot deal with this system because of the nonpolynomial components.

## 4 Conclusion

An abstract system has been constructed by choosing the appropriate predicates and discretizing the continuous state space of a nonpolynomial dynamical system. The abstract transitions have been automatically obtained using MetiTarski. The resulting finite state transition system can be sent to a model checker for verification purposes.

One important open question is concerned with choosing good predicates. Tiwari's method uses predicates that are constructed by taking repeated derivatives of the vector fields. The motivation being that for polynomial systems this process terminates. For nonpolynomial systems this is not necessarily the case. It will be necessary to quantify the *quality* of the abstractions. One potential option to increase the quality of the generated abstractions is to use the Counter Example Guided Abstraction Refinement (CEGAR) framework.

Barrier Certificates can be used for safety analysis [7] and are another source of good predicates. Instead of being concerned with the stability of an equilibrium point, they are used to prove that certain states of a system cannot be reached. This is done using Lyapunov theory (see Definition 5). The important point is that SOSTOOLS can be used to search for Barrier Certificates. Linear hybrid systems have been successfully verified using these techniques [8]. Future work includes using MetiTarski for determining abstract transitions of systems discretized by Barrier Certificates.

## References

1 B. Akbarpour and L. C. Paulson. Applications of MetiTarski in the verification of control and hybrid systems. In *Hybrid Systems: Computation and Control*, volume 5469 of *Lecture Notes in Computer Science*, pages 1–15, 2009.

2 Behzad Akbarpour and Lawrence C. Paulson. MetiTarski: An automatic theorem prover for real-valued special functions. *Journal of Automated Reasoning*, 44:175–205, 2010.

3 W. Denman, B. Akbarpour, S. Tahar, M. H. Zaki, and L. C. Paulson. Formal verification of analog designs using MetiTarski. In *Formal Methods in Computer-Aided Design. FMCAD 2009*, pages 93 –100, November 2009.

4 Goran Frehse, Colas Le Guernic, Alexandre Donzé, Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang, and Oded Maler. SpaceEx: Scalable verification of hybrid systems. In *Proc. 23rd International Conference on Computer Aided Verification (CAV)*, LNCS. Springer, 2011.

5 K Furuta, M Yamakita, and S Kobayashi. Swing-up control of inverted pendulum using pseudo-state feedback. *Part I: Journal of Systems and Control Engineering*, 206:263–269, 1992.

6 Emmanuel Hainry. Reachability in linear dynamical systems. In *Proceedings of the 4th conference on Computability in Europe: Logic and Theory of Algorithms*, CiE ’08, pages 241–250, Berlin, Heidelberg, 2008. Springer-Verlag.

7 Stephen Prajna. Barrier certificates for nonlinear model validation. *Automatica*, 42(1):117–126, 2006.

8 Stephen Prajna and Ali Jadbabaie. Safety verification of hybrid systems using barrier certificates. In *In Hybrid Systems: Computation and Control*, pages 477–492. Springer, 2004.

9 Stephen Prajna, Antonis Papachristodoulou, Peter Seiler, and Pablo A. Parrilo. SOSTOOLS: Sum of squares optimization toolbox for MATLAB, 2004.

10 Shankar Sastry. *Nonlinear Systems.* Springer, 1999.

11 Edward R. Scheinerman. *Invitation to Dynamical systems.* Dover Publications, 1996.

12 C. Sloth and R. Wisniewski. Abstraction of continuous dynamical systems utilizing lyapunov functions. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 3760–3765, December 2010.

13 Alfred Tarski. A decision method for elementary algebra and geometry. Technical report, RAND Corp., 1948.

14 Ashish Tiwari and Gaurav Khanna. Series of abstractions for hybrid automata. In *Hybrid Systems: Computation and Control HSCC*, volume 2289 of *LNCS*, pages 465–478. Springer, March 2002.