NLTOOLBOX: A library for reachability computation of nonlinear dynamical systems

Romain Testylier and Thao Dang

VERIMAG/CNRS, 2 avenue de Vignate, 38610 Gières, France

Abstract. We describe NLTOOLBOX, a library of data structures and algorithms for reachability computation of nonlinear dynamical systems. It provides the users with an easy way to "program" their own analysis procedures or to solve other problems beyond verification. We illustrate the use of the library for the analysis of a biological model.

1 Introduction

Reachability analysis is a fundamental problem in model checking, program analysis, controller synthesis. This problem was initially motivated by the interest in extending model checking to hybrid systems (comprising both discrete and continuous dynamics). In addition, the behaviors of these systems are often nondeterministic due to various uncertainties which could be inherent or epistemic (such as unknown initial conditions, parameter values, multiple mode switchings). Reachability analysis involves computing the set of all possible trajectories under such uncertainties. There are numerous tools for reachable set computation, such as Checkmate [5], d/dt [1], MPT tool [11], level set toolbox [13] HySAT/iSAT [8], Ariadne [7], SpaceEx [9], Flow* [6]. Compared to the scalability of the existing techniques on linear systems, their scalability on nonlinear systems is much lower, not only because of their inherently higher complexity, but also because they often require sophisticated fine tuning of computation parameters (such as time steps, error tolerance), choice of set representations and exploration strategies. An automatic fine tuning can hardly be efficient for all types of systems, since it cannot include a-priori knowledge that the user possesses and a-posterior knowledge that he could gain from the analysis. It is thus important to provide the user with a possibility of "programming" the analysis process so that he can easily readjust the computation parameters or include exploration intention. For this reason, NLTOOLBOX[1] was designed as a $C++$ library providing an algorithmic infrastructure for reachability computation with which the user can write a simple $C++$ program to develop and explore different exploration strategies or to solve specific analysis problems. Two major functionalities of the library are: reachability analysis of polynomial systems (using the Bernstein expansion technique) and reachability analysis of general nonlinear systems (using hybridization). The latter can be applied directly to continuous-time systems while the former only to discrete-time systems and thus its use

[1] http://www-verimag.imag.fr/PEOPLE/Thao.Dang/nltoolboxlib.

for continuous-time systems requires a system time-discretization. The rest of the paper is organized as follows. We first present the main data structures and algorithms and then illustrate the use of the library on a biological model.

2 Data structures and reachability algorithms

Polytopes defined by constraints are the main set representation, which contains additional constructors for *template polyhedra*, *hyper-rectangles* and *hyper-octagons*. The library contains a number of set operations needed by reachability analysis, such as inclusion test, affine transformation, set splitting (used for refinement). In the following, we describe only two main reachability algorithms: one is based on hybridization [3] and the other on the Bernstein expansion [4]. The library also includes an algorithm specialized for multi-affine systems [14].

Reachability algorithm using the Bernstein expansion. This algorithm computes the reachable set (represented by template polyhedra) of a discrete-time polynomial systems $x[k+1] = \pi(x[k])$ from an initial polyhedron $P \subset \mathbb{R}^n$. To handle continuous-time systems, the library offers a number of discretization methods. For a given template matrix T, we need to find a vector b such that the image $\pi(P)$ is included in the template polyhedron defined by $Tx \leq b$. To determine b, we formulate an polynomial optimization problems and replace it by a linear program (which can be solved more efficiently) by using affine bound functions. To compute affine bound functions for polynomials, the Bernstein expansion can be used. Indeed, an n-variate polynomial can be represented in the Bernstein basis functions and the coefficients of this representation allow capturing geometric properties of the polynomial and thus obtaining accurate function approximations. However, the Bernstein expansion is valid only inside the unit box $[0,1]^n$, and to address this problem, we use two methods: (1) oriented-box approximation and (2) rewriting the polynomial using a change of variables. Furthermore, two methods for handling templates are used: the template can be *static* (the polyhedra share the same constant matrix T) or *dynamic* (the matrix T evolves according to a local approximation of the dynamics).

Reachability algorithm using hybridization. The main idea of hybridization is to approximate a nonlinear system $\dot{x} = f(x)$ by a piecewise affine one. We compute an approximation domain (that contains the current reachable set) and an approximate vector field for that domain. When the system leaves the current approximation domain, a new domain is created. Our hybridization algorithm uses simplicial domains and piecewise affine approximate vector fields, which is motivated by many available methods for piecewise affine systems (see for instance [1, 5, 11, 10, 9]). In addition, we exploit the curvature of the vector field f to determine large domains with good error bound. To handle resulting piecewise affine systems, the library includes a basic reachability algorithm [1].

Programming an analysis procedure. These algorithms were successfully applied to many case studies (in particular a mitochondrial aging model with 9 variables and a model of ongiogenesis with 12 variables) and they were also evaluated using randomly generated systems, which shows that they can handle

efficiently systems with up to 10 variables and are among the state-of-the-art computational methods for nonlinear systems (see [3, 2]). The goal of this section is to demonstrate the usefulness of NLTOOLBOX by showing how to program with the library to solve a reachability problem. As a working example, we use the Laub-Loomis model [12] for spontaneous oscillations during the aggregation stage of Dictyostelium [12]: $\dot{x} = f(x)$, where the state variable $x = (x_0, \ldots, x_6)$ represents the concentrations of seven proteins, and the derivatives are $f_0 = k_1x_2 - k_2x_0$, $f_1 = k_3x_4 - k_4x_1$, $f_2 = k_5x_6 - k_6x_2x_1$, $f_3 = k_7 - k_8x_3x_2$, $f_4 = k_9x_0 - k_{10}x_3x_4$, $f_5 = k_{11}x_0 - k_{12}x_5$, $f_6 = k_{13}x_5 - k_{14}x_6x_1$. The model has 14 parameters $(k_1, \ldots, k_{14})$. The main steps of an analysis procedure using hybridization is shown in the following (pseudo) C++ program.

```
void Dictyostelium_hybridization() {
  1: createOctagonalSet(n, r, c, T, b); Hpolyhedron I(n, T, b);
  2: PointerSystem Sp(n, fp, df, hp);
  3: ReachHybridization reachHyb(Sp, I, err, dt);
  4: reachHybridization.reach(nbIter);
  5: vector<Hpolyhedron> res=reachHyb.getReachabilityResult();
  6: exporter.save(res, color);
}
```

In line 1 we define the octogonal initial set I, centered at c, with circumradius r. In line 2, we create a dynamical system Sp by specifying the pointers to the functions computing f, the Jacobian matrix of f and the Hessian matrix of f (used to define curvature). In line 3, an instance $reachHyb$ of the class *ReachHybridization* is created with a desired error bound err and a time step dt. Then, the reachable set is computed for $nbIter$ iterations and stored in res. The final phase (line 6) involves saving the result in a matlab file for visualization purposes. It is possible to choose the templates for the viewing projection (by default the box templates are used). To use the Bernstein technique, an instance $reachBern$ of the class *ReachPolynomial* can be created. Figure 1 shows the reachability results for the initial octagonal set with circumradius 0.0001 cented at $(1.2, 1.105, 1.5, 2.4, 1.0, 0.1, 0.45)$. For the hybridization method, the time step is 0.007 and the computation time for 4000 iterations is $346.43s$ (on a machine with 2.2 GHz Intel Core 2 Duo Processor). For the Berstein technique, the time step is 0.028, and the computation time for 1000 iterations is $283.46s$. The hybridization technique for this example is less time-efficient but more accurate when the reachable set converges towards the attraction basin.

3 Conclusion

The advantage of the library is twofold. On one hand, it provides the users with an easy way to "program" their own analysis procedures or to use reachability algorithms to solve other problems beyond verification. On the other hand, the library can also be used by other existing tools to increase their scope in terms of problems and methods. NLTOOLBOX is currently being integrated in SpaceEx [9], to extend the applicability of SpaceEx to nonlinear hybrid systems. Our future work includes using the library to for controller synthesis.

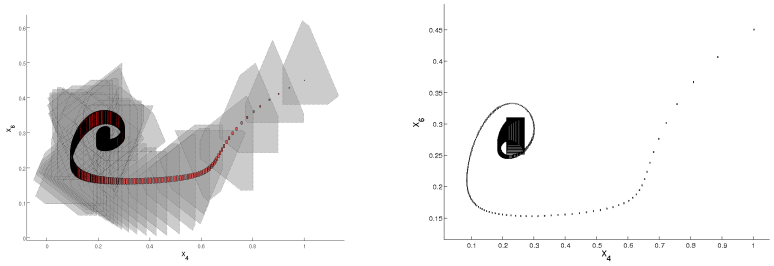

Fig. 1. The reachable sets computed by hybridization (left) and by the Bernstein technique (right). The grey areas in the left figure are the template projections of the hybridization domains.

References

1. E. Asarin, O. Bournez, T. Dang, and O. Maler. Approximate reachability analysis of piecewise linear dynamical systems. In *HSCC'00*, LNCS 1790, 21-31, 2000.
2. T. Dang, C. Le Guernic, and O. Maler. Computing reachable states for nonlinear biological models. *Theoretical Computer Science* 412(21):2095-2107, 2011.
3. T. Dang and R. Testylier. Hybridization domain construction using curvature estimation. In *HSCC'11*, LNCS, pp 123-132, 2011.
4. T. Dang and R. Testylier. Reachability analysis for polynomial dynamical systems using the Bernstein expansion. *Reliable Computing Journal*, ISSN 1573-1340, 2011.
5. A. Chutinan and B.H. Krogh. Computational Techniques for Hybrid System Verification. *IEEE Trans. on Automatic Control* **48**, 64-75, 2003.
6. X. Chen, E. Ábrahám, and S. Sankaranarayanan. Taylor model flowpipe construction for non-linear hybrid systems. In *Proc. RTSS12*. IEEE, 2012.
7. L. Benvenuti, D. Bresolin, A. Casagrande, P. Collins, A. Ferrari, E. Mazzi, A. Sangiovanni-Vincentelli, and R. Villa. Reachability computation for hybrid systems with Ariadne. In *Proc. the 17th IFAC World Congress*, 2008.
8. M. Fränzle, C. Herde, T. Teige, S. Ratschan, and T. Schubert. Efficient Solving of Large Non-Linear Arithmetic Constraint Systems with Complex Boolean Structure. *J. on Satisfiability, Boolean Modeling, and Computation*, Vol 1 (2007), pp 209-236.
9. G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. SpaceEx: Scalable verification of hybrid systems. In Proc. CAV11, LNCS 6806, pp 379395. Springer, 2011.
10. A. Kurzhanskiy and P. Varaiya, Ellipsoidal Techniques for Reachability Analysis of Discrete-time Linear Systems, *IEEE Trans. Automatic Control* **52**, 26-38, 2007.
11. M. Kvasnica, P. Grieder, M. Baotic, and Manfred Morari. Multi-Parametric Toolbox (MPT) In *HSCC'04*, LNCS 2993, pp 448-462, Springer, 2004.
12. M. T. Laub and W. F. Loomis. A Molecular Network That Produces Spontaneous Oscillations in Excitable Cells of Dictyostelium. *Molecular Biology of the Cell*, Vol 9, 3521-3532, 1998.
13. I. Mitchell and J. A. Templeton. A Toolbox of Hamilton-Jacobi Solvers for Analysis of Nondeterministic Continuous and Hybrid Systems. In *HSCC'05*, LNCS 3414, pp.480-494, 2005.
14. R. Testylier and T. Dang. Analysis of Parametric Biological Models. *Workshop on Hybrid Systems and Biology HSB 2012*, Springer, 2012.