Nonlinear Analysis: Hybrid Systems 27 (2018) 437–451

Contents lists available at ScienceDirect

Nonlinear Analysis: Hybrid Systems

journal homepage: www.elsevier.com/locate/nahs

Compositional abstraction refinement for control synthesis☆

Pierre-Jean Meyer *, Dimos V. Dimarogonas

ACCESS Linnaeus Center, School of Electrical Engineering, KTH Royal Institute of Technology, SE-100 44, Stockholm, Sweden

ARTICLE INFO

Article history:
Received 7 July 2016
Accepted 10 October 2017
Available online 13 November 2017

Keywords:
Symbolic control
Abstraction refinement
Compositional synthesis
Hybrid systems

ABSTRACT

This paper presents a compositional approach to specification-guided abstraction refinement for control synthesis of a nonlinear system associated with a method to over-approximate its reachable sets. Given an initial coarse partition of the state space, the control specification is given as a sequence of the cells of this partition to visit at each sampling time. The dynamics are decomposed into subsystems where some states and inputs are not observed, some states are observed but not controlled and where assume-guarantee obligations are used on the uncontrolled states of each subsystem. A finite abstraction is created for each subsystem through a refinement procedure starting from a coarse partition of the state space, then proceeding backwards on the specification sequence to iteratively split the elements of the partition whose coarseness prevents the satisfaction of the specification. Each refined abstraction is associated with a controller and it is proved that combining these local controllers can enforce the specification on the original system. The efficiency of the proposed approach compared to other abstraction methods is illustrated in a numerical example.

1. Introduction

In the past decades, a lot of work has been devoted towards model checking and plan synthesis of a finite transition system with respect to high-level specifications such as Linear Temporal Logic [1]. However, when the system is too large to be handled by such methods in reasonable time or when the system is not finite (e.g. continuous dynamics), we must rely on abstraction methods that create a smaller finite system related to the concrete system through a behavioral relationship such as simulation, bisimulation or their alternating and approximate versions [2]. Despite the significant progress in both the fields of model checking and abstraction, the link between them is not as straightforward as it appears. For example, it is possible that the specification is satisfied on the concrete system but not on the chosen abstraction, which would thus require looking for a finer abstraction where this satisfaction can be proved.

This observation is the origin of the development of an interface layer named *abstraction refinement*, whose goal is to use both the dynamics and the specification to automatically obtain an abstract system satisfying the specification by iteratively refining an initial coarse abstraction. This topic has received many contributions, mainly during the 1990s and early 2000s in the context of model checking for hardware design. As a consequence, these works primarily focus on verification (as opposed to control synthesis) of a formula in fragments of Computation Tree Logic (CTL) for large but finite systems, where the abstraction procedure is thus only used to reduce the complexity. The most popular approach is called *CounterExample-Guided Abstraction Refinement* (CEGAR) and consists in exploiting the counterexample provided

☆ This work was supported by the H2020 ERC Starting Grant BUCOPHSYS, the EU H2020 AEROWORKS project, the EU H2020 Co4Robots project, the Swedish Foundation for Strategic Research, the Swedish Research Council and the KAW Foundation.

* Corresponding author.
E-mail addresses: pjmeyer@kth.se (P. Meyer), dimos@kth.se (D.V. Dimarogonas).

https://doi.org/10.1016/j.nahs.2017.10.006
1751-570X/

by the model checker when the abstraction does not satisfy the formula in order to see where the abstraction is too coarse. These counterexamples can either guide the refinement towards splitting the discrete states of the abstraction where the counterexample originates [3] or improving the partial description of a decomposable system by considering more subsystems [4–6]. Other refinement methods also consider computing under- or over-approximations of the concrete transition system with iteratively increasing accuracy [7–9], using reachability analysis in a bisimulation algorithm [10] or its formula-guided version [11], or uniformly splitting the cells of the state partition based on some error measurement for stochastic systems [12].

In this paper, we present a method for specification-guided abstraction refinement for control synthesis of continuous systems. In this approach, a coarse abstraction of the system is initially considered and iteratively refined (through repartitioning of the state space) in its elements preventing the satisfaction of the specification. The problem of abstraction refinement for control synthesis has been mostly unexplored by the methods mentioned in the previous paragraph due to the fact that counterexamples of control problems are much harder to use to guide the refinement since they take the form of trees [13,14] (instead of single paths for model checking). Our approach instead considers a specification-guided approach using reachability analysis to identify the elements of the abstraction that need to be refined in order to find a satisfying controller. In addition, although some of the previously mentioned works consider infinite state space [10,11,13,15], most results of the above initial literature on abstraction refinement assume that the forward or backward reachable sets can be exactly computed, which is rarely true in systems with a continuous state space. As a consequence, approaches based on reachability analysis to split *good* and *bad* states into two disjoint sets [3,10,11,13,14] lose a part of their efficiency in such cases. Our approach thus relies on methods to efficiently compute over-approximations of reachable sets, using for example polytopes [16], oriented hyper-rectangles [17], ellipsoids [18], zonotopes [19], level sets [20] or the monotonicity property [21]. The use of such over-approximations ensures that a controller synthesized on the abstraction can be applied to the original system in order to satisfy the same specification. The recent years have seen a renewed interest on the topic of abstraction refinement, but with a focus on control synthesis for continuous systems as in the present paper. Among the most relevant work, we can see several refinement approaches applied to different types of abstraction. Indeed, while we use over-approximations of finite-time reachable sets to obtain a non-deterministic abstraction, other approaches consider infinite-time reachability analysis [22], using some feedback controller on the continuous system to obtain a deterministic abstraction [23], or using sets of finite prefixes to describe abstractions of infinite behaviors [24]. Another abstraction refinement approach is given in [25], where the refinement is not guided by the specifications as in our approach but by some behavioral relationship (similar to approximate bisimulation) which is not satisfied on the initial coarse abstraction. Another relevant method is [26], where the refinement approach is applied on an automaton structure related to the specifications instead of on an abstraction of the system as we do.

As any abstraction-based verification and synthesis problem, this approach is limited to low-dimensional systems due to the classical exponential growth of the abstraction size when the dimension of the state space increases. This paper thus aims at introducing this abstraction refinement method within a compositional approach where a control objective for the whole system is achieved by working on smaller components [27], thus widening the range of applications to systems of larger dimensions or systems only equipped with distributed computation capabilities (e.g. multi-robot systems). More specifically, we adapt the compositional abstraction method presented in [28] and [29] to this abstraction refinement framework. In this approach, the global system is decomposed into subsystems representing partial descriptions of the dynamics, where some of the states and inputs are not observed and some states are observed to increase the precision of the model but not controlled. Then, an abstraction can be created for each subsystem using the abstraction refinement approach and the composition of the controllers synthesized on each of these abstractions can be used to control the original system. To reduce the conservatism of this compositional approach, we consider an *assume-guarantee* reasoning [30], similarly used in e.g., the recent results [31] to synthesize controlled invariant sets and [32] for a symbolic control synthesis using small-gain theorem. With such reasoning, the abstraction of each subsystem is obtained under the assumption that other subsystems satisfy their own control objectives and the controller synthesized on one subsystem is then used to guarantee that the assumptions of other subsystems hold. Defining abstraction refinement within a compositional framework has been mostly unexplored in the literature apart from some results on finite systems [33–36] and, to the best of our knowledge, a single contribution on systems with infinite state space (using hybrid automata) [37]. As opposed to these papers which are all based on the CEGAR method and thus rely on a model checker providing counterexamples to guide the abstraction refinement, our approach only uses reachability analysis in order to detect unsatisfiability of the specification.

The structure of this paper is as follows. The problem is formulated in Section 2. Section 3 describes the general method to obtain compositional abstractions. The abstraction refinement algorithm to be applied to each subsystem is presented in Section 4. Then, Section 5 provides the main result that the local controllers can be composed to control the original system. Finally, a numerical illustration of this method for the temperature regulation in a 8-room building is presented in Section 6.

2. Problem formulation

2.1. Notations

In this paper, a decomposition of a system into subsystems is considered. As a result, both scalar and set variables are used as subscript of other variables, sets or functions: