Toolbox for Symbolic Control from Hybrid Automata Specifications

Vladimir Sinyakov and Antoine Girard[*][†]

April 19, 2020

1 Overview

The toolbox provides methods for solving control synthesis problems for discrete and continuous dynamical systems with uncertainty. The dynamics is described by one of the following equations:

$$\dot{x} = f(x, u, w) \tag{1}$$

or

$$x_{k+1} = f(x_k, u_k, w_k). \tag{2}$$

The toolbox contains the algorithms for abstraction of those systems.

Specifications of the desired system behavior are given by discrete-time hybrid automata:

$$(x_{k+1}, p_{k+1}) \in G(x_k, v_k). \tag{3}$$

with a terminal condition $(x_k, p_k, v_kl) \in W_f$. The sampled system obtained from (1) or the system (2) must alternatingly simulate the specification (3) as well as satisfy the specified reachability or safety requirement (see [1]). The toolbox can also work with specifications given by control programs which were introduced in [2]. The preprints of the mentioned papers can be found in the */help* folder.

2 Installation and system requirements

The toolbox has been tested on MATLAB R2017b and MATLAB R2019a. To run the algorithms, add the root folder to the paths list or set it as a working folder.

3 Algorithms

```
function [TS, TransitionNumber] = AbstractSystemDiscrete(F, Partition, Controls, Disturbance)
```

For a given discrete-time mixed monotone system S, given partitions of the state and control spaces and a given disturbance set, the function computes the abstract system which is alternatingly simulated by S.

```
function [TS, TransitionNumber] = AbstractSystemContinuous(F, Partition, Controls, Disturbance)
```

For a given discrete-time mixed monotone system S, given partitions of the state and control spaces and a given disturbance set, the function computes the abstract system which is alternatingly simulated by S.

```
function TS_with_Spec = AbstractSpecificationDiscrete(Spec, Partition, TS, TransitionNumber)
```

For a given discrete-time hybrid automata, a given partition of the state space and a precomputed abstract system, the function computes the abstract specification system which contains only transitions allowed by the precomputed abstraction. Parameter $TransitionNumber$ contains the information about the number of transitions in the precomputed abstraction corresponding to a particular state and a particular control input.

[*]V. Sinyakov and A. Girard are with Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des Signaux et Systèmes, 91190, Gif-sur-Yvette, France. E-mail: `firstname.lastname@l2s.centralesupelec.fr`

[†]This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 725144).

```
function [V, C, domC] = ComputeControllerReachability(TS, X1)
```

For a given abstract system and a given set of terminal states, the function computes the controller C which solves the reachability problem. Vector V defines the value function which is the number of transitions needed to reach the terminal set, $domC$ is the list of controllable states.

```
function [V, C, domC] = ComputeControllerSafety(TS, X1)
```

For a given abstract system and a given set of terminal states, the function computes the controller C which solves the safety problem with this terminal set. Vector V defines the value function which is equal to $+\infty$ if the state is uncontrollable, to 0 if the state is in the invariant safe set, and to 1 if the state is not in the invariant safe set but the terminal set is reachable. $domC$ is the list of controllable states.

```
function [V, C] = ControlProgram(TS, Tasks, R, W0, algorithm)
```

For an abstract system, task specifications, scheduler, program terminal set, and a chosen program synthesis algorithm, the function computes the collection of task controllers and the respective collection of value functions.

4 Examples

There are currently three examples which, with minimal modifications, reproduce the calculations and results of Examples 1 and 2 from [1] and of the control program synthesis example from [2]. To obtain the proper results and pictures, the code blocks in the respective scripts should be executed sequentially.

References

[1] V. Sinyakov and A. Girard. Formal Controller Synthesis from Specifications Given by Discrete-Time Hybrid Automata. preprint, https://hal.archives-ouvertes.fr/hal-02361404, Nov. 2019.

[2] V. Sinyakov and A. Girard. Formal synthesis from control programs. 2020.