Approximate Bisimulation and Discretization of Hybrid CSP

Gaogao Yan, Li Jiao, Yangjia Li, Shuling Wang$^{(\boxtimes)}$, and Naijun Zhan$^{(\boxtimes)}$

State Key Laboratory of Computer Science, Institute of Software,
Chinese Academy of Sciences, Beijing, China
{yangg,ljiao,yangjia,wangsl,znj}@ios.ac.cn

Abstract. Hybrid Communicating Sequential Processes (HCSP) is a powerful formal modeling language for hybrid systems, which is an extension of CSP by introducing differential equations for modeling continuous evolution and interrupts for modeling interaction between continuous and discrete dynamics. In this paper, we investigate the semantic foundation for HCSP from an operational point of view by proposing the notion of approximate bisimulation, which provides an appropriate criterion to characterize the equivalence between HCSP processes with continuous and discrete behaviour. We give an algorithm to determine whether two HCSP processes are approximately bisimilar. In addition, based on which, we propose an approach on how to discretize HCSP, i.e., given an HCSP process A, we construct another HCSP process B which does not contain any continuous dynamics such that A and B are approximately bisimilar with given precisions. This provides a rigorous way to transform a verified control model to a correct program model, which fills the gap in the design of embedded systems.

Keywords: HCSP · Approximately bisimilar · Hybrid systems · Discretization

1 Introduction

Embedded Systems (ESs) make use of computer units to control physical processes so that the behavior of the controlled processes meets expected requirements. They have become ubiquitous in our daily life, e.g., automotive, aerospace, consumer electronics, communications, medical, manufacturing and so on. ESs are used to carry out highly complex and often critical functions such as to monitor and control industrial plants, complex transportation equipments, communication infrastructure, etc. The development process of ESs is widely recognized as a highly complex and challenging task. Model-Based Engineering

This work is supported partly by "973 Program" under grant No. 2014CB340701, by NSFC under grants 91418204 and 61502467, by CDZ project CAP (GZ 1023), and by the CAS/SAFEA International Partnership Program for Creative Research Teams.

© Springer International Publishing AG 2016

J. Fitzgerald et al. (Eds.): FM 2016, LNCS 9995, pp. 702–720, 2016.
DOI: 10.1007/978-3-319-48989-6_43

(MBE) is considered as an effective way of developing correct complex ESs, and has been successfully applied in industry [16,21]. In the framework of MBE, a model of the system to be developed is defined at the beginning; then extensive analysis and verification are conducted based on the model so that errors can be detected and corrected at early stages of design of the system. Afterwards, model transformation techniques are applied to transform abstract formal models into more concrete models, even into source code.

To improve the efficiency and reliability of MBE, it is absolutely necessary to automate the system design process as much as possible. This requires that all models at different abstraction levels have a precise mathematical semantics. Transformation between models at different abstraction levels should preserve semantics, which can be done automatically with tool support.

Thus, the first challenge in model-based formal design of ESs is to have a powerful modelling language which can model all kinds of features of ESs such as communication, synchronization, concurrency, continuous and discrete dynamics and their interaction, real-time, and so on, in an easy way. To address this issue, Hybrid Communicating Sequential Processes (HCSP) was proposed in [14,36], which is an extension of CSP by introducing differential equations for modeling continuous evolutions and interrupts for modeling interaction between continuous and discrete dynamics. Comparing with other formalisms, e.g., hybrid automata [17], hybrid programs [24], etc., HCSP is more expressive and much easier to be used, as it provides a rich set of constructors. Through which a complicated ES with different behaviours can be easily modeled in a compositional way. The semantic foundation of HCSP has been investigated in the literature, e.g., in He's original work on HCSP [14], an algebraic semantics of HCSP was given by defining a set of algebraic laws for the constructors of HCSP. Subsequently, a DC-based semantics for HCSP was presented in [36] due to Zhou *et al.* These two original formal semantics of HCSP are very restrictive and incomplete, for example, it is unclear whether the set of algebraic rules defined in [14] is complete, and super-dense computation and recursion are not well handled in [36]. In [8,13,22,33,35], the axiomatic, operational, and the DC-based and UTP-based denotational semantics for HCSP are proposed, and the relations among them are discussed. However, regarding operational semantics, just a set of transition rules was proposed in [35]. It is unclear in what sense two HCSP processes are equivalent from an operational point of view, which is the cornerstone of operational semantics, also the basis of refinement theory for a process algebra. So, it absolutely deserves to investigate the semantic foundation of HCSP from an operational point of view.

Another challenge in the model-based formal design of ESs is how to transform higher level abstract models (control models) to lower level program models (algorithm models), even to C code, seamlessly in a rigorous way. Although huge volume of model-based development approaches targeting embedded systems has been proposed and used in industry and academia, e.g., Simulink/Stateflow [1,2], SCADE [9], Modelica [31], SysML [3], MARTE [28], Ptolemy [10], hybrid automata [17], CHARON [5], HCSP [14,36], Differential Dynamic Logic [24], and Hybrid Hoare Logic [22], the gap between higher-level control models and lower-level algorithm models still remains.

Approximate bisimulation [12] is a popular method for analyzing and verifying complex hybrid systems. Instead of requiring observational behaviors of two systems to be exactly identical, it allows errors but requires the "distance" between two systems remain bounded by some precisions. In [11], with the use of simulation functions, a characterization of approximate simulation relations between hybrid systems is developed. A new approximate bisimulation relation with two parameters as precisions, which is very similar to the notion defined in this paper, is introduced in [18]. For control systems with inputs, the method for constructing a symbolic model which is approximately bisimilar with the original continuous system is studied in [26]. Moreover, [23] discusses the problem for building an approximately bisimilar symbolic model of a digital control system. Also, there are some works on building symbolic models for networks of control systems [27]. But for all the above works, either discrete dynamics is not considered, or it is assumed to be atomic actions independent of the continuous variables. In [15,20,32], the abstraction of hybrid automata is considered, but it is only guaranteed that the abstract system is an approximate simulation of the original system. In [25], a discretization of hybrid programs is presented for a proof-theoretical purpose, i.e., it aims to have a sound and complete axiomatization relative to properties of discrete programs. Differently from all the above works, we aim to have a discretization of HCSP, for which discrete and continuous dynamics, communications, and so on, are entangled with each other tightly, to guarantee that the discretized process has the approximate equivalence with the original process.

The main contributions of this paper include:

- First of all, we propose the notion of approximate bisimulation, which provides a criterion to characterize in what sense two HCSP processes with differential kinds of behaviours are equivalent from an operational point of view. Based on which, a refinement theory for HCSP could be developed.
- Then, we show that whether two HCSP processes are approximately bisimilar or not is decidable if all ordinary differential equations (ODEs) occurring in them satisfy globally asymptotical stability (GAS) condition (the definition will be given later). This is achieved by proposing an algorithm to compute an approximate bisimulation relation for the two HCSP processes.
- Most importantly, we present how to discretize an HCSP process (a control model) by a discrete HCSP process (an algorithm model), and prove they are approximately bisimilar, if the original HCSP process satisfies the GAS condition and is robustly safe with respect to some given precisions.

The rest of this paper is organized as follows: In Sect. 2, we introduce some preliminary notions on dynamical systems. Sect. 3 defines transition systems and the approximate bisimulation relation between transition systems. The syntax and the transition semantics of HCSP, and the approximately bisimilar of HCSP processes are presented in Sect. 4. The discretization of HCSP is presented in Sect. 5. Throughout the paper, and in Sect. 6, a case study on the water tank system [4] is shown to illustrate our method. At the end, Sect. 7 concludes the paper and discusses the future work. For space limitation, the proofs for all the lemmas and theorems are omitted, but can be found in [34].

2 Preliminary

In this section, we briefly review some notions in dynamical systems, that can be found at [19,29]. In what follows, $\mathbb{N}$, $\mathbb{R}$, $\mathbb{R}^+$, $\mathbb{R}_0^+$ denote the natural, real, positive and nonnegative real numbers, respectively. Given a vector $\mathbf{x} \in \mathbb{R}^n$, $\|\mathbf{x}\|$ denotes the infinity norm of $\mathbf{x} \in \mathbb{R}^n$, i.e., $\|\mathbf{x}\| = \max\{|x_1|, |x_2|, ..., |x_n|\}$. A continuous function $\gamma : \mathbb{R}_0^+ \rightarrow \mathbb{R}_0^+$, is said in class $\mathcal{K}$ if it is strictly increasing and $\gamma(0) = 0$; γ is said in class $\mathcal{K}_\infty$ if $\gamma \in \mathcal{K}$ and $\gamma(r) \rightarrow \infty$ as $r \rightarrow \infty$. A continuous function $\beta : \mathbb{R}_0^+ \times \mathbb{R}_0^+ \rightarrow \mathbb{R}_0^+$ is said in class $\mathcal{KL}$ if for each fixed s, the map $\beta(r, s) \in \mathcal{K}_\infty$ with respect to r and, for each fixed r, $\beta(r, s)$ is decreasing with respect to s and $\beta(r, s) \rightarrow 0$ as $s \rightarrow \infty$.

A dynamical system is of the following form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \qquad \mathbf{x}(t_0) = \mathbf{x}_0 \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state and $\mathbf{x}(t_0) = \mathbf{x}_0$ is the *initial condition.*

Suppose $a < t_0 < b$. A function $X(.) : (a, b) \rightarrow \mathbb{R}^n$ is said to be a *trajectory* (solution) of (1) on (a, b), if $X(t_0) = \mathbf{x}_0$ and $\dot{X}(t) = \mathbf{f}(X(t))$ for all $t \geq t_0$. In order to ensure the existence and uniqueness of trajectories, we assume $\mathbf{f}$ satisfying the local Lipschitz condition, i.e., for every compact set $S \subset \mathbb{R}^n$, there exists a constant $L > 0$ s.t. $\|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$, for all $\mathbf{x}, \mathbf{y} \in S$. Then, we write $X(t, \mathbf{x}_0)$ to denote the point reached at time $t \in (a, b)$ from initial condition $\mathbf{x}_0$, which should be uniquely determined. In addition, we assume (1) is *forward complete* [7], i.e., it is solvable on an open interval $(a, +\infty)$. An equilibrium point of (1) is a point $\bar{\mathbf{x}} \in \mathbb{R}^n$ s.t. $\mathbf{f}(\bar{\mathbf{x}}) = 0$.

Definition 1. *A dynamical system of form (1) is said to be* globally asymptotically stable *(*GAS*) if there exists a point* $\mathbf{x}_0$ *and a function* β *of class* $\mathcal{KL}$ *s.t.*

$$\forall \mathbf{x} \in \mathbb{R}^n \ \forall t \geq 0. \|X(t, \mathbf{x}) - \mathbf{x}_0\| \leq \beta(\|\mathbf{x} - \mathbf{x}_0\|, t).$$

It is easy to see that the point $\mathbf{x}_0$ is actually the unique equilibrium point of the system. When this point is previously known or can be easily computed, one can prove the system to be GAS by constructing a corresponding Lyapunov function. However, $\mathbf{x}_0$ cannot be found sometimes, for example, when the dynamics $\mathbf{f}$ of the system depends on external inputs and thus is not completely known. The concept of δ-GAS would be useful in this case.

Definition 2. *A dynamical system of (1) is said to be incrementally globally asymptotically stable (*δ-GAS*) if it is forward complete and there is a* $\mathcal{KL}$ *function* β *s.t.*

$$\forall \mathbf{x} \in \mathbb{R}^n \ \forall \mathbf{y} \in \mathbb{R}^n \ \forall t \geq 0. \|X(t, \mathbf{x}) - X(t, \mathbf{y})\| \leq \beta(\|\mathbf{x} - \mathbf{y}\|, t).$$

In [6], the relationship between GAS and δ-GAS was established, restated by the following proposition.

Proposition 1. – *If (1) is* δ-GAS, *then it is* GAS.

– *If there exist two strictly positive reals M and ε, and a differentiable function $V(\mathbf{x}, \mathbf{y})$ with $\alpha_1(\|\mathbf{x} - \mathbf{y}\|) \le V(\mathbf{x}, \mathbf{y}) \le \alpha_2(\|\mathbf{x} - \mathbf{y}\|)$ for some α_1, α_2 and ρ of class $\mathcal{K}_\infty$, s.t.*

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n. \begin{pmatrix} \|\mathbf{x} - \mathbf{y}\| \le \varepsilon \wedge \|\mathbf{x}\| \ge M \wedge \|\mathbf{y}\| \ge M \\ \Rightarrow \frac{\partial V}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}) + \frac{\partial V}{\partial \mathbf{y}} \mathbf{f}(\mathbf{y}) \le -\rho(\|\mathbf{x} - \mathbf{y}\|) \end{pmatrix},$$

then the system (1) is δ-GAS.

A function $V(\mathbf{x}, \mathbf{y})$ satisfying the condition in Proposition 1 is called a δ-GAS *Lyapunov function* of (1). Proposition 1 tells us that (1) is δ-GAS if and only if it admits a δ-GAS Lyapunov function. In general, checking the inequality in Definition 2 is difficult, one may construct δ-GAS Lyapunov functions as an alternative.

3 Transition Systems and Approximate Bisimulation

In the following, the set of actions, denoted by *Act*, is assumed to consist of a set of discrete actions which take no time (written as $\mathcal{E}$), $\mathbb{R}_0^+$ the set of delay actions which just take time delay, and a special internal action τ. Actions are ranged over $l_1, \ldots, l_n, \ldots$.

Definition 3 (Transition system). *A labeled transition system with observations is a tuple $T = \langle Q, L, \rightarrow, Q^0, Y, H \rangle$, where Q is a set of states, $L \subseteq Act$ is a set of labels, $Q^0 \subseteq Q$ is a set of initial states, Y is a set of observations, and H is an observation function $H : Q \rightarrow Y$, $\rightarrow \subseteq Q \times L \times Q$ is a transition relation, satisfying*

*1, **identity:** $q \xrightarrow{0} q$ always holds;*

*2, **delay determinism:** if $q \xrightarrow{d} q'$ and $q \xrightarrow{d} q''$, then $q' = q''$; and*

*3, **delay additivity:** if $q \xrightarrow{d_1} q'$ and $q' \xrightarrow{d_2} q''$ then $q \xrightarrow{d_1+d_2} q''$, where d, d_1, $d_2 \in \mathbb{R}_0^+$.*

A transition system T is said to be *symbolic* if Q and $L \cap \mathcal{E}$ are finite, and $L \cap \mathbb{R}_0^+$ is bounded, and *metric* if the output set Y is equipped with a metric $\mathbf{d} : Y \times Y \rightarrow \mathbb{R}_0^+$. In this paper, we regard Y as being equipped with the metric $\mathbf{d}(\mathbf{y}_1, \mathbf{y}_2) = \|\mathbf{y}_1 - \mathbf{y}_2\|$.

A *state trajectory* of a transition system T is a (possibly infinite) sequence of transitions $\mathbf{q}^0 \xrightarrow{l^0} \mathbf{q}^1 \xrightarrow{l^1} \cdots \xrightarrow{l^{i-1}} \mathbf{q}^i \xrightarrow{l^i} \cdots$, denoted by $\{\mathbf{q}^i \xrightarrow{l^i} \mathbf{q}^{i+1}\}_{i \in \mathbb{N}}$, s.t. $\mathbf{q}^0 \in Q^0$ and for any i, $\mathbf{q}^i \xrightarrow{l^i} \mathbf{q}^{i+1}$. An *observation trajectory* is a (possibly infinite) sequence $\mathbf{y}^0 \xrightarrow{l^0} \mathbf{y}^1 \xrightarrow{l^1} \cdots \xrightarrow{l^{i-1}} \mathbf{y}^i \xrightarrow{l^i} \cdots$, denoted by $\{\mathbf{y}^i \xrightarrow{l^i} \mathbf{y}^{i+1}\}_{i \in \mathbb{N}}$, and it is accepted by T if there exists a corresponding state trajectory of T s.t. $\mathbf{y}^i = H(\mathbf{q}^i)$ for any $i \in \mathbb{N}$. The set of observation trajectories accepted by T is called the *language* of T, and is denoted by $L(T)$. The reachable set of T is a subset of Y defined by

$$Reach(T) = \{\mathbf{y} \in Y | \exists \{\mathbf{y}^i \xrightarrow{l^i} \mathbf{y}^{i+1}\}_{i \in \mathbb{N}} \in L(T), \exists j \in \mathbb{N}, \mathbf{y}^j = \mathbf{y}\}.$$

We can verify the safety property of T by computing $Reach(T) \cap Y_U$, in which $Y_U \subseteq Y$ is the set of unsafe observations. If it is empty, then T is *safe*, otherwise, *unsafe*.

For a maximum sequence of τ actions $\mathbf{q}^i \xrightarrow{\tau} \mathbf{q}^{i+1} \xrightarrow{\tau} \cdots \xrightarrow{\tau} \mathbf{q}^{i+k}$, we remove the intermediate states and define the τ-*compressed* transition $\mathbf{q}^i \overset{\tau}{\twoheadrightarrow} \mathbf{q}^{i+k}$ instead. For unification, for a non-τ transition $\mathbf{q}^i \xrightarrow{l^i} \mathbf{q}^{i+1}$ where $l^i \neq \tau$, we define $\mathbf{q}^i \overset{l^i}{\twoheadrightarrow} \mathbf{q}^{i+1}$. In what follows, we will denote $\langle Q, L, \twoheadrightarrow, Q^0, Y, H\rangle$ the resulting labeled transition system from $\langle Q, L, \rightarrow, Q^0, Y, H\rangle$ by replacing each label transition with its τ-compressed version. As a common convention in process algebra, we use $\mathbf{p} \overset{l}{\Longrightarrow} \mathbf{p}'$ to denote the closure of τ transitions, i.e., $\mathbf{p}(\overset{\tau}{\twoheadrightarrow})^{\{0,1\}} \overset{l}{\twoheadrightarrow} (\overset{\tau}{\twoheadrightarrow})^{\{0,1\}}\mathbf{p}'$, for any $l \in L$ in the sequel.

Given $l_1, l_2 \in L \cup \{\tau\}$, we define the *distance* $dis(l_1, l_2)$ between them as follows:

$$dis(l_1, l_2) \overset{\text{def}}{=} \begin{cases} 0 & \text{if both } l_1 \text{ and } l_2 \text{ are in } \mathcal{E} \text{ or are } \tau \\ |d - d'| & \text{if } l_1 = d \text{ and } l_2 = d' \text{ are both delay actions, i.e., } d, d' \in \mathbb{R}_0^+ \\ \infty & \text{Otherwise} \end{cases}$$

Definition 4 (Approximate bisimulation). *Let $T_i = \langle Q_i, L_i, \twoheadrightarrow_i, Q_i^0, Y_i, H_i\rangle$, $(i = 1, 2)$ be two metric transition systems with the same output set Y and metric $\mathbf{d}$. Let h and ε be the time and value precision respectively. A relation $\mathcal{B}_{h,\varepsilon} \subseteq Q_1 \times Q_2$ is called a (h, ε)-approximate bisimulation relation between T_1 and T_2, if for all $(\mathbf{q}_1, \mathbf{q}_2) \in \mathcal{B}_{h,\varepsilon}$,*

1. $\mathbf{d}(H_1(\mathbf{q}_1), H_2(\mathbf{q}_2)) \leq \varepsilon$,
2. $\forall \mathbf{q}_1 \overset{l}{\twoheadrightarrow}_1 \mathbf{q}_1'$, $\exists \mathbf{q}_2 \overset{l'}{\Longrightarrow}_2 \mathbf{q}_2'$ *s.t.* $dis(l, l') \leq h$ *and* $(\mathbf{q}_1', \mathbf{q}_2') \in \mathcal{B}_{h,\varepsilon}$, *for* $l \in L_1$ *and* $l' \in L_2$
3. $\forall \mathbf{q}_2 \overset{l}{\twoheadrightarrow}_2 \mathbf{q}_2'$, $\exists \mathbf{q}_1 \overset{l'}{\Longrightarrow}_1 \mathbf{q}_1'$ *s.t.* $dis(l, l') \leq h$ *and* $(\mathbf{q}_1', \mathbf{q}_2') \in \mathcal{B}_{h,\varepsilon}$, *for* $l \in L_2$ *and* $l' \in L_1$.

Definition 5. *T_1 and T_2 are approximately bisimilar with the precision h and ε (denoted $T_1 \cong_{h,\varepsilon} T_2$), if there exists a (h, ε)-approximate bisimulation relation $\mathcal{B}_{h,\varepsilon}$ between T_1 and T_2 s.t. for all $\mathbf{q}_1 \in Q_1^0$, there exists $\mathbf{q}_2 \in Q_2^0$ s.t. $(\mathbf{q}_1, \mathbf{q}_2) \in \mathcal{B}_{h,\varepsilon}$, and vice versa.*

The following result ensures that the set of (h, ε)-approximate bisimulation relations has a maximal element.

Lemma 1. *Let*$\{\mathcal{B}^i_{h,\varepsilon}\}_{i\in I}$ *be a family of* (h, ε)*-approximate bisimulation relations between* T_1 *and* T_2*. Then,* $\bigcup_{i\in I} \mathcal{B}^i_{h,\varepsilon}$ *is a* (h, ε)*-approximate bisimulation relation between* T_1 *and* T_2*.*

By Lemma 1, given the precision parameters h and ε, let $\{\mathcal{B}^i_{h,\varepsilon}\}_{i\in I}$ be the set of all (h, ε)-approximate bisimulation relations between T_1 and T_2, then the maximal (h, ε)-approximate bisimulation relation between T_1 and T_2 is defined by $\mathcal{B}^{max}_{h,\varepsilon} = \bigcup\limits_{i\in I} \mathcal{B}^i_{h,\varepsilon}$. For two transition systems that are approximately bisimilar, the reachable sets have the following relationship:

Theorem 1. *If* $T_1 \cong_{h,\varepsilon} T_2$*, then* $\mathit{Reach}(T_1) \subseteq N(\mathit{Reach}(T_2), \varepsilon)$*, where* $N(Y, \varepsilon)$ *denotes the* ε *neighborhood of* Y*, i.e.* $\{x \mid \exists y.y \in Y \wedge \|x - y\| < \varepsilon\}$*.*

Thus, if the distance between $\mathit{Reach}(T_2)$ and the unsafe set Y_U is greater than ε, then the intersection of $\mathit{Reach}(T_1)$ and Y_U is empty and hence T_1 is safe, whenever $T_1 \cong_{h,\varepsilon} T_2$.

4 Hybrid CSP (HCSP)

In this section, we present a brief introduction to HCSP and define the transition system of HCSP from an operational point of view. An example is given for better understanding. Finally, we investigate the approximate bisimilarity for HCSP processes.

4.1 HCSP

Hybrid Communicating Sequential Process (HCSP) is a formal language for describing hybrid systems, which extends CSP by introducing differential equations for modelling continuous evolutions and interrupts for modeling the arbitrary interaction between continuous evolutions and discrete jumps. The syntax of HCSP can be described as follows:

$$\begin{aligned} P ::= &\,\mathrm{skip} \mid x := e \mid \mathrm{wait}\ d \mid ch?x \mid ch!e \mid P; Q \mid B \rightarrow P \mid P \sqcap Q \mid P^* \\ &\mid []_{i\in I} io_i \rightarrow P_i \mid \langle F(\dot{\mathrm{s}}, \mathrm{s}) = 0 \& B\rangle \mid \langle F(\dot{\mathrm{s}}, \mathrm{s}) = 0 \& B\rangle \trianglerighteq []_{i\in I}(io_i \rightarrow Q_i) \\ S ::= &\,P \mid S\|S \end{aligned}$$

where x, s for variables and vectors of variables, respectively, B and e are boolean and arithmetic expressions, d is a non-negative real constant, ch is the channel name, io_i stands for a communication event, i.e., either $ch_i?x$ or $ch_i!e$, P, Q, Q_i are sequential process terms, and S stands for an HCSP process term. Given an HCSP process S, we define $\mathit{Var}(S)$ for the set of variables in S, and $\Sigma(S)$ the set of channels occurring in S, respectively. The informal meanings of the individual constructors are as follows:

- skip, $x := e$, wait d, $ch?x$, $ch!e$, $P;Q$, $P \sqcap Q$, and $[]_{i\in I} io_i \rightarrow P_i$ are defined as usual. $B \rightarrow P$ behaves as P if B is true, otherwise terminates.
- For repetition P^*, P executes for an arbitrary finite number of times. We assume an oracle *num*, s.t. for a given P^* in the context process S, $num(P^*, S)$ returns the upper bound of the number of times that P is repeated in the context.
- $\langle F(\dot{\mathbf{s}}, \mathbf{s}) = 0 \& B\rangle$ is the continuous evolution statement. It forces the vector $\mathbf{s}$ of real variables to obey the differential equations F as long as B, which defines the domain of $\mathbf{s}$, holds, and terminates when B turns false. Without loss of generality, we assume that the set of B is open, thus the escaping point will be at the boundary of B. The communication interrupt $\langle F(\dot{\mathbf{s}}, \mathbf{s}) = 0 \& B\rangle \trianglerighteq []_{i\in I}(io_i \rightarrow Q_i)$ behaves like $\langle F(\dot{\mathbf{s}}, \mathbf{s}) = 0 \& B\rangle$, except that the continuous evolution is preempted as soon as one of the communications io_i takes place, which is followed by the respective Q_i. These two statements are the main extension of HCSP for describing continuous behavior.
- $S_1 \| S_2$ behaves as if S_1 and S_2 run independently except that all communications along the common channels connecting S_1 and S_2 are to be synchronized. S_1 and S_2 in parallel can neither share variables, nor input or output channels.

For better understanding of the HCSP syntax, we model the water tank system [4], for which two components *Watertank* and *Controller*, are composed in parallel. The HCSP model of the system is given by *WTS* as follows:

$$\begin{aligned}
&\mathit{WTS} &&\overset{\text{def}}{=} \mathit{Watertank} \| \mathit{Controller}\\
&\mathit{Watertank} &&\overset{\text{def}}{=} v := v_0; d := d_0;\\
& &&(v = 1 \rightarrow \langle \dot{d} = Q_{max} - \pi r^2 \sqrt{2gd}\rangle \trianglerighteq (wl!d \rightarrow cv?v);\\
& &&v = 0 \rightarrow \langle \dot{d} = -\pi r^2 \sqrt{2gd}\rangle \trianglerighteq (wl!d \rightarrow cv?v))^*\\
&\mathit{Controller} &&\overset{\text{def}}{=} y := v_0; x := d_0; (\text{wait } p; wl?x; x \geq ub \rightarrow y := 0;\\
& &&x \leq lb \rightarrow y := 1; cv!y)^*
\end{aligned}$$

where Q_{max}, π, r and g are system parameters, v is the control variable which takes 1 or 0, depending on whether the valve is open or not, d is the water level of the *Watertank* and its dynamics depends on the value of v. v_0 and d_0 are the initial values of controller variable and water level, respectively. Two channels, *wl* and *cv*, are used to transfer the water level (d in *Watertank*) and control variable (y in *Controller*) between *Watertank* and *Controller*, respectively. The control value is computed by the *Controller* with a period of p. When the water level is less than or equal to *lb*, the control value is assigned to 1, and when the water level is greater than or equal to *ub*, the control value is assigned to 0, otherwise, it keeps unchanged. Basically, based on the current value of v, *Watertank* and *Controller* run independently for p time, then *Watertank* sends the current water level to *Controller*, according to which a new value of the control variable is generated and sent back to *Watertank*, after that, a new period repeats.

4.2 Transition System of HCSP

Given an HCSP process S, we can derive a transition system $T(S) = \langle Q, L, \rightarrow, Q^0, Y, H\rangle$ from S by the following procedure:

- the set of states $Q = (\mathit{subp}(S) \cup \{\epsilon\}) \times V(S)$, where $\mathit{subp}(S)$ is the set of sub-processes of S, e.g., $\mathit{subp}(S) = \{S, \text{wait } d, B \rightarrow P\} \cup \mathit{subp}(P)$ for $S ::= \text{wait } d; B \rightarrow P$, ϵ is introduced to represent the terminal process, meaning that the process has terminated, and $V(S) = \{v | v \in \mathit{Var}(S) \rightarrow \mathit{Val}\}$ is the set of evaluations of the variables in S, with Val representing the value space of variables. Without confusion in the context, we often call an evaluation v a (process) state. Given a state $q \in Q$, we will use $\mathit{fst}(q)$ and $\mathit{snd}(q)$ to return the first and second component of q, respectively.
- The label set L corresponds to the actions of HCSP, defined as $L = \mathbb{R}_0^+ \cup \Sigma(S) \,.\, \{?, !\} \,.\, \mathbb{R} \cup \{\tau\}$, where $d \in \mathbb{R}_0^+$ stands for the time progress, $ch?c, ch!c \in \Sigma(S) \,.\, \{?, !\} \,.\, \mathbb{R}$ means that an input along channel ch with value c being received, an output along ch with value c being sent, respectively. Besides, the silent action τ represents a discrete non-communication action of HCSP, such as assignment, evaluation of boolean expressions, and so on.
- $Q^0 = \{(S, v) | v \in V(S)\}$, representing that S has not started to execute, and v is the initial process state of S.
- $Y = \overline{\mathit{Val}}$, represents the set of value vectors corresponding to $\mathit{Var}(S)$.
- Given $q \in Q$, $H(q) = \mathit{vec}(\mathit{snd}(q))$, where function vec returns the value vector corresponding to the process state of q.
- $\rightarrow$ is the transition relation of S, which is given next.

Sequential Processes. A transition relation of a sequential HCSP process takes the form $(P, v) \xrightarrow{l} (P', v')$, indicating that starting from state v, P executes to P' by performing action l, with the resulting state v'. Here we present the transition rules for continuous evolution as an illustration. Readers are referred to [35] for the full details of the transition semantics, for both sequential and parallel HCSP processes.

$$\frac{\begin{array}{c}\forall d > 0. \exists S(.) : [0, d] \rightarrow \mathbb{R}^n.(S(0) = v(\mathbf{s}) \wedge (\forall p \in [0, d).(F(\dot{S}(p), S(p)) = 0 \\ \wedge v[\mathbf{s} \mapsto S(p)](B) = \mathit{true})))\end{array}}{(\langle F(\dot{\mathbf{s}}, \mathbf{s}) = 0 \& B\rangle, v) \xrightarrow{d} (\langle F(\dot{\mathbf{s}}, \mathbf{s}) = 0 \& B\rangle, v[\mathbf{s} \mapsto S(d)])}$$

$$\frac{v(B) = false}{(\langle F(\dot{\mathbf{s}}, \mathbf{s}) = 0 \& B\rangle, v) \xrightarrow{\tau} (\epsilon, v)}$$

For $\langle F(\dot{\mathbf{s}}, \mathbf{s}) = 0 \& B\rangle$, for any $d \geq 0$, it evolves for d time units according to F if B evaluates to true within this period (the right end exclusive). In the rule, $S(\cdot) : [0, d] \rightarrow \mathbb{R}^n$ defines the trajectory of the ODE F with initial value $v(\mathbf{s})$. Otherwise, by performing a τ action, the continuous evolution terminates if B evaluates to false.

Parallel Composition. Given two sequential processes P_1, P_2 and their transition systems $T(P_1) = \langle Q_1, L_1, \rightarrow_1, Q_1^0, Y_1, H_1\rangle$ and $T(P_2) = \langle Q_2, L_2, \rightarrow_2, Q_2^0, Y_2, H_2\rangle$, we can define the transition system of $P_1\|P_2$ as $T(P_1\|P_2) = \langle Q, L, \rightarrow, Q, Y, H\rangle$, where:

- $Q = ((subp(P_1) \cup \{\epsilon\})\|(subp(P_2) \cup \{\epsilon\})) \times \{v_1 \uplus v_2 | v_1 \in V(P_1), v_2 \in V(P_2)\}$, where given two sets of processes PS_1 and PS_2, $PS_1\|PS_2$ is defined as $\{\alpha\|\beta | \alpha \in PS_1 \wedge \beta \in PS_2\}$; $v_1 \uplus v_2$ represents the disjoint union, i.e. $v_1 \uplus v_2(x)$ is $v_1(x)$ if $x \in Var(P_1)$, otherwise $v_2(x)$.
- $L = L_1 \cup L_2$.
- $Q^0 = \{(P_1\|P_2, v_1^0 \uplus v_2^0) | (P_i, v_i^0) \in Q_i^0 \text{ for } i = 1, 2\}$.
- $Y = Y_1 \times Y_2$, the observation space of the parallel composition is obviously the Cartesian product of Y_1 and Y_2.
- $H(q) = H_1(q) \times H_2(q)$, the observation function is the Cartesian product of the two component observation functions correspondingly.
- $\rightarrow$ is defined based on the parallel composition of transitions of L_1 and L_2.

Suppose two transitions $(P_1, u) \xrightarrow{\alpha} (P_1', u')$ and $(P_2, v) \xrightarrow{\beta} (P_2', v')$ occur for P_1 and P_2, respectively. The rule for synchronization is given below:

$$\frac{\alpha = ch_i?c \wedge \beta = ch_i!e \wedge c = e}{(P_1\|P_2, u \uplus v) \xrightarrow{\tau} (P_1'\|P_2', u' \uplus v')}$$

4.3 Approximate Bisimulation Between HCSP Processes

Let P_1 and P_2 be two HCSP processes, and h, ε the time and value precisions. Let v_0 be an arbitrary initial state. P_1 and P_2 are (h, ε)-*approximately bisimilar*, denoted by $P_1 \cong_{h,\varepsilon} P_2$, if $T(P_1) \cong_{h,\varepsilon} T(P_2)$, in which $T(P_1)$ and $T(P_2)$ are the τ-compressed transition systems of P_1 and P_2 with the same initial state v_0, respectively.

In Algorithm 1, we consider the (h, ε)-approximate bisimilation between P_1 and P_2 for which all the ODEs occurring in P_1 and P_2 are GAS. Suppose the set of ODEs occurring in P_i is $\{F_1^i, \cdots, F_{ki}^i\}$, and the equilibrium points for them are $x_1^i, \cdots, x_{ki}^i$ for $i = 1, 2$ respectively. As a result, for each ODE, there must exist a sufficiently large time, called *equilibrium time*, s.t. after the time, the distance between the trajectory and the equilibrium point is less than ε. We denote the equilibrium time for each F_j^i for $j = 1, \cdots, ki$ by T_j^i, respectively. Furthermore, in order to record the execution time of ODEs, for each ODE F_j^i, we introduce an auxiliary time variable t_j^i and add $t_j^i := 0; \dot{t_j^i} = 1$ to F_j^i correspondingly.

Algorithm 1 decides whether P_1 and P_2 are (h, ε)-approximately bisimilar. When $P_1 \cong_{h,\varepsilon} P_2$, it returns **true**, otherwise, it returns **false**. Let d be the discretized time step. The algorithm is then taken in two steps. The first step (lines 1–6) constructs the transition systems for P_1 and P_2 with time step d. For $m = 1, 2$, $T(P_m).Q$ and $T(P_m).T$ represent the reachable set of states and transitions of P_m, respectively, which are initialized as empty sets and then constructed iteratively. At each step i, a new transition can be a d time progress,

Algorithm 1. Deciding approximately bisimilar between two HCSP processes

```
Input: Processes $P_1, P_2$, the initial state $v_0$, the time step $d$, and precisions $h$ and $\varepsilon$;
Initialization:
    $T(P_m).Q^0 = \{(P_m, v_0)\}, T(P_m).T^0 = \emptyset$ for $m = 1, 2$; $i = 0$;
 1: repeat
 2:     $T(P_m).T^{i+1} = T(P_m).T^i \cup \{q \overset{l}{\twoheadrightarrow} q' | \forall q \in T(P_m).Q^i$, if $(\exists l \in \{d, \tau\} \cup \Sigma(P_m) \centerdot \{?, !\} \centerdot \mathbb{R}. q \overset{l}{\twoheadrightarrow} q')$ or $(\exists l = d'. l < d \wedge q \overset{l}{\twoheadrightarrow} q' \wedge$ not $(q \overset{d''}{\twoheadrightarrow})$ for any $d''$ in $(d', d])$ and $snd(q')(t_j^m) < T_j^m\}$;
 3:     $T(P_m).Q^{i+1} = T(P_m).Q^i \cup postState(T(P_m).T^{i+1})$;
 4:     $i \leftarrow i + 1$;
 5: until $T(P_m).T^i = T(P_m).T^{i-1}$
 6: $T(P_m).Q = T(P_m).Q^i; T(P_m).T = T(P_m).T^i$;
 7: $\mathcal{B}^0_{h,\varepsilon} = \{(q_1, q_2) \in T(P_1).Q \times T(P_2).Q | \mathbf{d}(H_1(q_1), H_2(q_2)) \le \varepsilon\}$; $i = 0$;
 8: repeat
 9:     $\mathcal{B}^{i+1}_{h,\varepsilon} \leftarrow \{(q_1, q_2) \in B^i_{h,\varepsilon} | \forall q_1 \overset{l}{\twoheadrightarrow}_1 q_1' \in T(P_1).T, \exists q_2 \overset{l'}{\Longrightarrow}_2 q_2' \in T(P_2).T$ s.t. $(q_1', q_2') \in \mathcal{B}^i_{h,\varepsilon}$ and $dis(l, l') \le h$, and $\forall q_2 \overset{l}{\twoheadrightarrow}_2 q_2' \in T(P_2).T, \exists q_1 \overset{l'}{\Longrightarrow}_1 q_1' \in T(P_1).T$ s.t. $(q_1', q_2') \in \mathcal{B}^i_{h,\varepsilon}$ and $dis(l, l') \le h\}$;
10:     $i \leftarrow i + 1$;
11: until $\mathcal{B}^i_{h,\varepsilon} = \mathcal{B}^{i-1}_{h,\varepsilon}$
12: $\mathcal{B}_{h,\varepsilon} = \mathcal{B}^i_{h,\varepsilon}$;
13: if $((P_1, v_0), (P_2, v_0)) \in \mathcal{B}_{h,\varepsilon}$ then
14:     return true;
15: else
16:     return false;
17: end if
```

a τ event, or a communication event. Besides, a transition can be a time progress less than d, which might be caused by the occurrence of a boundary interrupt or a communication interrupt during a continuous evolution. The new transition will be added only when the running time for each ODE F_j^m, denoted by t_j^m, is less than the corresponding equilibrium time. Therefore, for either process P_m, whenever some ODE runs beyond its equilibrium time, the set of reachable transitions reaches a fixpoint by allowing precision ε and will not be extended any more. The set of reachable states can be obtained by collecting the post states of reachable transitions. Based on Definition 4, the second step (lines 7–17) decides whether the transition systems for P_1 and P_2 are approximately bisimilar with the given precisions.

The first part (lines 1–6) of the algorithm computes the transitions of processes. For each process P_m, its complexity is $O(|T(P_m).T|)$, which is $O(\lceil \frac{T_m}{d} \rceil + N_m)$, where T_m represents the execution time of P_m till termination or reaching the equilibrium time of some ODE, and N_m the number of atomic statements of P_m. The second part (lines 7–17) checks for P_1 and P_2 each pair of the states whose distance is within ε by traversing the outgoing transitions, to see if they are truly approximate bisimilar, till the fixpoint $\mathcal{B}_{h,\varepsilon}$ is reached.

We can compute the time complexity to be $O(Q_1^2 Q_2^2 T_1 T_2)$, where Q_m and T_m represent $O(|T(P_m).Q|)$ and $O(|T(P_m).T|)$ for $m = 1, 2$ respectively.

Theorem 2 (Correctness). *Algorithm 1 terminates, and for any v_0, $P_1 \cong_{h,\varepsilon} P_2$ iff $((P_1, v_0), (P_2, v_0)) \in \mathcal{B}_{h,\varepsilon}$.*

5 Discretization of HCSP

In this section, we consider the discretization of HCSP processes, by which the continuous dynamics is represented by discrete approximation. Let P be an HCSP process and (h, ε) be the precisions, our goal is to construct a discrete process D from P, s.t. P is (h, ε)-bisimilar with D, i.e., $P \cong_{h,\varepsilon} D$ holds.

5.1 Discretization of Continuous Dynamics

Since most differential equations do not have explicit solutions, the discretization of the dynamics is normally given by discrete approximation. Consider the ODE $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ with the initial value $\widetilde{\mathbf{x}}_0 \in \mathbb{R}^n$, and assume $X(t, \widetilde{\mathbf{x}}_0)$ is the trajectory of the initial value problem along the time interval $[t_0, \infty)$. In the following discretization, assume h and ξ represent the time step size and the precision of the discretization, respectively. Our strategy is as follows:

- First, from the fact that $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ is GAS, there must exist a sufficiently large T s.t. $\|X(t, \widetilde{\mathbf{x}}_0) - \bar{\mathbf{x}}\| < \xi$ holds when $t > T$, where $\bar{\mathbf{x}}$ is an equilibrium point. As a result, after time T, the value of $\mathbf{x}$ can be approximated by the equilibrium point $\bar{\mathbf{x}}$ and the distance between the actual value of $\mathbf{x}$ and $\bar{\mathbf{x}}$ is always within ξ.
- Then, for the bounded time interval $[t_0, T]$, we apply Euler method to discretize the continuous dynamics.

There are a range of different discretization methods for ODEs [30] and the Euler method is an effective one among them. According to the Euler method, the ODE $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ is discretized as

$$(\mathbf{x} := \mathbf{x} + h\mathbf{f}(\mathbf{x}); \text{wait } h)^N$$

A sequence of approximate solutions $\{\mathbf{x}_i\}$ at time stamps $\{h_i\}$ for $i = 1, 2, \cdots, N$ with $N = \lceil \frac{T - t_0}{h} \rceil$ are obtained, satisfying (define $\mathbf{x}_0 = \widetilde{\mathbf{x}}_0$):

$$h_i = t_0 + i * h \quad \mathbf{x}_i = \mathbf{x}_{i-1} + h\mathbf{f}(\mathbf{x}_{i-1}).$$

$\|X(h_i, \widetilde{\mathbf{x}}_0) - \mathbf{x}_i\|$ represents the discretization error at time h_i. To estimate the global error of the approximation, by Theorem **3** in [25], we can prove the following theorem:

Theorem 3 (Global error with an initial error). *Let $X(t, \widetilde{\mathbf{x}}_0)$ be a solution on $[t_0, T]$ of the initial value problem $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \mathbf{x}(t_0) = \widetilde{\mathbf{x}}_0$, and L the Lipschitz constant s.t. for any compact set S of $\mathbb{R}^n$, $\|\mathbf{f}(\mathbf{y}_1) - \mathbf{f}(\mathbf{y}_2)\| \leq L\|\mathbf{y}_1 - \mathbf{y}_2\|$ for all $\mathbf{y}_1, \mathbf{y}_2 \in S$. Let $\mathbf{x}_0 \in \mathbb{R}^n$ satisfy $\|\mathbf{x}_0 - \widetilde{\mathbf{x}}_0\| \leq \xi_1$. Then there exists an $h_0 > 0$, s.t. for all h satisfying $0 < h \leq h_0$, and for all n satisfying $nh \leq (T - t_0)$, the sequence $\mathbf{x}_n = \mathbf{x}_{n-1} + h\mathbf{f}(\mathbf{x}_{n-1})$ satisfies:*

$$\|X(nh, \widetilde{\mathbf{x}}_0) - \mathbf{x}_n\| \leq e^{(T-t_0)L}\xi_1 + \frac{h}{2} \max_{\zeta \in [t_0, T]} \|X''(\zeta, \widetilde{\mathbf{x}}_0)\| \frac{e^{L(T-t_0)} - 1}{L}$$

By Theorem 3 and the property of GAS, we can prove the following main theorem.

Theorem 4 (Approximation of an ODE). *Let $X(t, \widetilde{\mathbf{x}}_0)$ be a solution on $[t_0, \infty]$ of the initial value problem $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \mathbf{x}(t_0) = \widetilde{\mathbf{x}}_0$, and L the Lipschitz constant. Assume $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ is GAS with the equilibrium point $\bar{\mathbf{x}}$. Then for any precision $\xi > 0$, there exist $h > 0, T > 0$ and $\xi_1 > 0$ s.t. $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \mathbf{x}(t_0) = \widetilde{\mathbf{x}}_0$ and $\mathbf{x} := \mathbf{x}_0; (\mathbf{x} := \mathbf{x} + h\mathbf{f}(\mathbf{x}); \textit{wait } h)^N; \mathbf{x} := \bar{\mathbf{x}}; \textbf{stop}$ with $N = \lceil \frac{T-t_0}{h} \rceil$ are (h, ξ)-approximately bisimilar, in which $\|\mathbf{x}_0 - \widetilde{\mathbf{x}}_0\| < \xi_1$ holds, i.e., there is an error between the initial values.*

5.2 Discretization of HCSP

We continue to consider the discretization of HCSP processes, among which any arbitrary number of ODEs, the discrete dynamics, and communications are involved. Below, given an HCSP process P, we use $D_{h,\varepsilon}(P)$ to represent the discretized process of P, with parameters h and ε to denote the step size and the precision (i.e. the maximal "distance" between states in P and $D_{h,\varepsilon}(P)$), respectively.

Before giving the discretization of HCSP processes, we need to introduce the notion of readiness variables. In order to express the readiness information of communication events, for each channel ch, we introduce two boolean variables $ch?$ and $ch!$, to represent whether the input and output events along ch are ready to occur. We will see that in the discretization, the readiness information of partner events is necessary to specify the behavior of communication interrupt.

Table 1 lists the definition of $D_{h,\varepsilon}(P)$. For each rule, the original process is listed above the line, while the discretized process is defined below the line. For skip, $x := e$ and wait d, they are kept unchanged in the discretization. For input $ch?x$, it is discretized as itself, and furthermore, before $ch?x$ occurs, $ch?$ is assigned to 1 to represent that $ch?x$ becomes ready, and in contrary, after $ch?x$ occurs, $ch?$ is reset to 0. The output $ch!e$ is handled similarly. The compound constructs, $P; Q$, $P \sqcap Q$, P^* and $P\|Q$ are discretized inductively according to their structure. For $B \to P$, B is still approximated to B and P is discretized inductively. For external choice $[\!]_{i \in I} io_i \to P_i$, the readiness variables io_i for all $i \in I$ are set to 1 at first, and after the choice is taken, all of them are reset to 0

Table 1. The rules for discretization of HCSP

$$\frac{\text{skip}}{\text{skip}} \quad \frac{x := e}{x := e} \quad \frac{\text{wait } d}{\text{wait } d}$$

$$\frac{ch?x}{ch? := 1; ch?x; ch? := 0} \quad \frac{ch!e}{ch! := 1; ch!e; ch! := 0}$$

$$\frac{P; Q}{D_{h,\varepsilon}(P); D_{h,\varepsilon}(Q)} \quad \frac{B \rightarrow P}{B \rightarrow D_{h,\varepsilon}(P)} \quad \frac{P \sqcap Q}{D_{h,\varepsilon}(P) \sqcap D_{h,\varepsilon}(Q)}$$

$$\frac{[\!]_{i \in I} io_i \rightarrow P_i}{\forall i \in I.io_i := 1; [\!]_{i \in I} io_i \rightarrow (\forall i \in I.io_i := 0; D_{h,\varepsilon}(P_i))}$$

$$\frac{\langle \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \& B \rangle}{(N(B,\varepsilon) \rightarrow (\mathbf{x} := \mathbf{x} + h\mathbf{f}(\mathbf{x}); \text{wait } h))^{\lceil \frac{T}{h} \rceil}; N(B,\varepsilon) \rightarrow (\mathbf{x} := \bar{\mathbf{x}}; \mathbf{stop})}$$

$$\frac{\langle \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \& B \rangle \unrhd [\!]_{i \in I}(io_i \rightarrow Q_i)}{\begin{array}{c} \forall i \in I.io_i := 1; (N(B,\varepsilon) \rightarrow \forall i \in I.io_i \wedge \neg \overline{io_i} \rightarrow (\mathbf{x} := \mathbf{x} + h\mathbf{f}(\mathbf{x}); \text{wait } h))^{\lceil \frac{T}{h} \rceil}; \\ \neg N(B,\varepsilon) \wedge \forall i \in I.io_i \wedge \neg \overline{io_i} \rightarrow \forall i \in I.io_i := 0; \\ \exists i.io_i \wedge \overline{io_i} \rightarrow ([\!]_{i \in I} io_i \rightarrow (\forall i \in I.io_i := 0; D_{h,\varepsilon}(Q_i))); \\ (N(B,\varepsilon) \wedge \forall i \in I.io_i \wedge \neg \overline{io_i}) \rightarrow (\mathbf{x} := \bar{\mathbf{x}}; \mathbf{stop}); \end{array}}$$

$$\frac{P^*}{(D_{h,\varepsilon}(P))^*} \quad \frac{P \| Q}{D_{h,\varepsilon}(P) \| D_{h,\varepsilon}(Q)}$$

and the corresponding process is discretized. Notice that because I is finite, the $\forall$ operator is defined as an abbreviation of the conjunction over I.

Given a boolean expression B and a precision ε, we define $N(B,\varepsilon)$ to be a boolean expression which holds in the ε-neighbourhood of B. For instance, if B is $x > 2$, then $N(B,\varepsilon)$ is $x > 2 - \varepsilon$. For a continuous evolution $\langle \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \& B \rangle$, under the premise that $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ is GAS, there must exists time T such that when the time is larger than T, the distance between the actual state of $\mathbf{x}$ and the equilibrium point, denoted by $\bar{\mathbf{x}}$, is less than ε. Then according to Theorem 3, $\langle \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \& B \rangle$ is discretized as follows: First, it is a repetition of the assignment to $\mathbf{x}$ according to the Euler method for at most $\lceil \frac{T}{h} \rceil$ number of times, and then followed by the assignment of $\mathbf{x}$ to the equilibrium point and stop forever. Both of them are guarded by the condition $N(B,\varepsilon)$. For a communication interrupt $\langle \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \& B \rangle \unrhd [\!]_{i \in I}(io_i \rightarrow Q_i)$, suppose T is sufficiently large s.t. when the time is larger than T, the distance between the actual state of $\mathbf{x}$ and the equilibrium point, denoted by $\bar{\mathbf{x}}$, is less than ε, and furthermore, if the interruption occurs, it must occur before T, and let $\overline{ch*}$ be the dual of $ch*$, e.g., if $ch* = ch?$, then $\overline{ch*} = ch!$ and vice versa. After all the readiness variables corresponding to $\{io_i\}_I$ are set to 1 at the beginning, the discretization is taken by the following steps: first, if $N(B,\varepsilon)$ holds and no communication among $\{io_i\}_{i \in I}$ is ready, it executes following the discretization of continuous evolution, for at most $\lceil \frac{T}{h} \rceil$ number of steps; then if $N(B,\varepsilon)$ turns false without any communication occurring, the whole process terminates and meanwhile the readiness variables are reset to 0; otherwise if some communications get ready, an external choice between these ready communications is taken, and then, the readiness variables are reset to 0 and the corresponding Q_i is followed; finally, if the communications never occur and the continuous evolution never terminates, the continuous variable is assigned to the equilibrium point and the time progresses forever. It should be noticed that, the readiness variables of the partner processes will be used to decide whether a communication is able to occur. They are shared between parallel processes, but will always be written by one side.

Consider the water tank system introduced in Sect. 4, by using the rules in Table 1, a discretized system $WTS_{h,\varepsilon}$ is obtained as follows:

$$\begin{array}{lll}
WTS_{h,\varepsilon} & \overset{\text{def}}{=} & Watertank_{h,\varepsilon} \| Controller_{h,\varepsilon} \\
Watertank_{h,\varepsilon} & \overset{\text{def}}{=} & v := v_0; d := d_0; (v = 1 \rightarrow (wl! := 1; \\
& & (wl! \wedge \neg wl? \rightarrow (d = d + h(Q_{max} - \pi r^2 \sqrt{2gd}); wait\ h;))^{\lceil \frac{T_1}{h} \rceil}; \\
& & wl! \wedge wl? \rightarrow (wl!d; wl! := 0; cv? := 1; cv?v; cv? := 0); \\
& & wl! \wedge \neg wl? \rightarrow (d = Q_{max}^2 / 2g\pi^2 r^4; \mathbf{stop})); \\
& & v = 0 \rightarrow (wl! := 1; \\
& & (wl! \wedge \neg wl? \rightarrow (d = d + h(-\pi r^2 \sqrt{2gd}); wait\ h;))^{\lceil \frac{T_2}{h} \rceil}; \\
& & wl! \wedge wl? \rightarrow (wl!d; wl! := 0; cv? := 1; cv?v; cv? := 0); \\
& & wl! \wedge \neg wl? \rightarrow (d = 0; \mathbf{stop})))^* \\
Controller_{h,\varepsilon} & \overset{\text{def}}{=} & y := v_0; x := d_0; (wait\ p; wl? := 1; wl?x; wl? := 0; \\
& & x \geq ub \rightarrow y := 0; x \leq lb \rightarrow y := 1; cv! := 1; cv!y; cv! := 0)^*
\end{array}$$

5.3 Properties

Before giving the main theorem, we introduce some notations. In order to keep the consistency between the behavior of an HCSP process and its discretized process, we introduce the notion of (δ, ϵ)-robustly safe. First, let ϕ denote a formula and ϵ a precision, define $N(\phi, -\epsilon)$ as the set $\{\mathbf{x} | \mathbf{x} \in \phi \wedge \forall \mathbf{y} \in \neg\phi. \|\mathbf{x} - \mathbf{y}\| > \epsilon\}$. Intuitively, when $\mathbf{x} \in N(\phi, -\epsilon)$, then $\mathbf{x}$ is inside ϕ and moreover the distance between it and the boundary of ϕ is greater than ϵ.

Definition 6 ((δ, ϵ)-robustly safe). *An HCSP process P is (δ, ϵ)-robustly safe, for a given initial state v_0, a time precision $\delta > 0$ and a value precision $\epsilon > 0$, if the following two conditions hold:*

- *for every continuous evolution $\langle \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \& B \rangle$ occurring in P, when P executes up to $\langle \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \& B \rangle$ at time t with state v, if $v(B) = false$, then there exists $\widehat{t} > t$ with $\widehat{t} - t < \delta$ s.t. for any σ satisfying $\mathbf{d}(\sigma, v[\mathbf{x} \mapsto X(\widehat{t}, \widetilde{\mathbf{x}}_0)]) < \epsilon$, $\sigma \in N(\neg B, -\epsilon)$, where $X(t, \widetilde{\mathbf{x}}_0)])$ is the solution of $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ with initial value $\widetilde{\mathbf{x}}_0 = v_0(\mathbf{x})$;*
- *for every alternative process $B \rightarrow P$ occurring in S, if B depends on continuous variables of P, then when P executes up to $B \rightarrow P$ at state v, $v \in N(B, -\epsilon)$ or $v \in N(\neg B, -\epsilon)$.*

As a result, when P is discretized with a time error less than δ and a value error less than ϵ, then P and its discretized process have the same control flow. The main theorem is given below.

Theorem 5. *Let P be an HCSP process and v_0 is the initial state. Assume P is (δ, ϵ)-robustly safe with respect to v_0. Let $0 < \varepsilon < \epsilon$ be a precision. If for any ODE $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ occurring in P, $\mathbf{f}$ is Lipschitz continuous and $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ is GAS with $\mathbf{f}(\bar{\mathbf{x}}) = 0$ for some $\bar{\mathbf{x}}$, then there exist $h > 0$ and the equilibrium time for each ODE F in P, $T_F > 0$, s.t. $P \cong_{h,\varepsilon} D_{h,\varepsilon}(P)$.*

We can compute that, the relation $\mathcal{L}\delta + Mh \leq \varepsilon$ holds for some constants $\mathcal{L}$ and M. Especially, $\mathcal{L}$ is the maximum value of the first derivative of $\mathbf{x}$ with respect to t. More details can be found in [34].

6 Case Study

In this section, we illustrate our method through the safety verification of the water tank system, *WTS*, that is introduced in Sect. 4. The safety property is to maintain the value of d within $[low, high]$, which needs to compute the reachable set of *WTS*. However, it is usually difficult because of the complexity of the system. Fortunately, the reachable set of the discretized $WTS_{h,\varepsilon}$ in Sect. 5 could be easily obtained. Therefore, we can verify the original system *WTS* through the discretized one, $WTS_{h,\varepsilon}$, as follows.

Table 2. The reachable set for different precisions

ε	h	$Reach(WTS_{h,\varepsilon})$	$Reach(WTS)$
0.2	0.2	[3.41, 6.5]	[3.21, 6.7]
0.1	0.05	[3.42, 6.47]	[3.32, 6.57]
0.05	0.01	[3.43, 6.46]	[3.38, 6.51]

In order to analyze the system, first of all, we set the values of parameters to $Q_{max} = 2.0$, $\pi = 3.14$, $r = 0.18$, $g = 9.8$, $p = 1$, $lb = 4.1$, $ub = 5.9$, $low = 3.3$, $high = 6.6$, $v_0 = 1$, and $d_0 = 4.5$ (units are omitted here). Then, by simulation, we compute the values of δ and ϵ as 0.5 and 0.24, s.t. *WTS* is (δ, ϵ)-robustly safe. By Theorem 5, for a given ε with $0 < \varepsilon < \epsilon$, since $\dot{d}$ and d are monotonic for both ODEs, we can compute a $h > 0$ s.t. $WTS \cong_{h,\varepsilon} WTS_{h,\varepsilon}$. For different values of ε and h, $Reach(WTS_{h,\varepsilon})$ could be computed, and then based on Theorem 1, we can obtain $Reach(WTS)$. Table 2 shows the results for different choices of ε and h. As seen from the results, when the values of precisions become smaller, $Reach(WTS_{h,\varepsilon})$ and $Reach(WTS)$ get closer and tighter. For the smaller precisions, i.e., $(\varepsilon = 0.1, h = 0.05)$ and $(\varepsilon = 0.05, h = 0.01)$, the safety property of the system is proved to be true. However, for $(\varepsilon = 0.2, h = 0.2)$, the safety property of the system can not be promised.

7 Conclusion

Approximate bisimulation is a useful notion for analyzing complex dynamic systems via simpler abstract systems. In this paper, we define the approximate bisimulation of hybrid systems modelled by HCSP, and present an algorithm for deciding whether two HCSP processes are approximately bisimilar. We have proved that if all the ODEs are GAS, then the algorithm terminates in a finite number of steps. Furthermore, we define the discretization of HCSP processes, by representing the continuous dynamics by Euler approximation. We have proved for an HCSP process that, if the process is robustly safe, and if each ODE occurring in the process is Lipschitz continuous and GAS, then there must exist a discretization of the original HCSP process such that they are approximate bisimilar with the given precisions. Thus, the results of analysis performed on the discrete system can be carried over into the original dynamic system, and vice versa. At the end, we illustrate our method by presenting the discretization of a water tank example. Note that GAS and robust safety are very restrictive from a theoretical point of view, but most of real applications satisfy these conditions in practice.

Regarding future work, we will focus on the implementation, in particular, the transformation from HCSP to ANSI-C. Moreover, it could be interesting to investigate approximate bisimularity with time bounds so that the assumptions of GAS and robust safety can be dropped. In addition, it deserves to investigate richer refinement theories for HCSP based on the notion of approximately bisimulation, although itself can be seen as a refinement relation as discussed in process algebra.

References

1. Simulink User's Guide (2013). http://www.mathworks.com/help/pdf_doc/simulink/sl_using.pdf
2. Stateflow User's Guide (2013). http://www.mathworks.com/help/pdf_doc/stateflow/sf_using.pdf
3. SysML V 1.4 Beta Specification (2013). http://www.omg.org/spec/SysML
4. Ahmad, E., Dong, Y., Wang, S., Zhan, N., Zou, L.: Adding formal meanings to AADL with hybrid annex. In: Lanese, I., Madelaine, E. (eds.) FACS 2014. LNCS, vol. 8997, pp. 228–247. Springer, Heidelberg (2015). doi:10.1007/978-3-319-15317-9_15
5. Alur, R., Dang, T., Esposito, J., Hur, Y., Ivancic, F., Kumar, V., Mishra, P., Pappas, G., Sokolsky, O.: Hierarchical modeling and analysis of embedded systems. Proc. IEEE **91**(1), 11–28 (2003)
6. Angeli, D., et al.: A Lyapunov approach to incremental stability properties. IEEE Trans. Autom. Control **47**(3), 410–421 (2002)
7. Angeli, D., Sontag, E.: Forward completeness, unboundedness observability, and their Lyapunov characterizations. Syst. Control Lett. **38**(4), 209–217 (1999)
8. Chen, M., Ravn, A., Wang, S., Yang, M., Zhan, N.: A two-way path between formal and informal design of embedded systems. In: UTP 2016. LNCS (2016)

9. Dormoy, F.: Scade 6: a model based solution for safety critical software development. ERTS **08**, 1–9 (2008)
10. Eker, J., Janneck, J., et al.: Taming heterogeneity - the Ptolemy approach. Proc. IEEE **91**(1), 127–144 (2003)
11. Girard, A., Julius, A., Pappas, G.: Approximate simulation relations for hybrid systems. Discrete Event Dyn. Syst. **18**(2), 163–179 (2008)
12. Girard, A., Pappas, G.: Approximation metrics for discrete and continuous systems. IEEE Trans. Autom. Control **52**(5), 782–798 (2007)
13. Guelev, D., Wang, S., Zhan, N.: Hoare-style reasoning about hybrid CSP in the duration calculus. Technical report ISCAS-SKLCS-13-01, Institute of Software, Chinese Academy of Sciences (2013)
14. He, J.: From CSP to hybrid systems. In: A Classical Mind, Essays in Honour of C.A.R. Hoare, pp. 171–189. Prentice Hall International (UK) Ltd. (1994)
15. Henzinger, T., Ho, P., Wong-Toi, H.: Algorithmic analysis of nonlinear hybrid systems. IEEE Trans. Autom. Control **43**(4), 540–554 (1998)
16. Henzinger, T.A., Sifakis, J.: The embedded systems design challenge. In: Misra, J., Nipkow, T., Sekerinski, E. (eds.) FM 2006. LNCS, vol. 4085, pp. 1–15. Springer, Heidelberg (2006). doi:10.1007/11813040_1
17. Henzinger, T.A.: The theory of hybrid automata. In: LICS 1996, pp. 278–292 (1996)
18. Julius, A., D'Innocenzo, A., Di Benedetto, M., Pappas, G.: Approximate equivalence and synchronization of metric transition systems. Syst. Control Lett. **58**(2), 94–101 (2009)
19. Khalil, H.K., Grizzle, J.W.: Nonlinear Systems, vol. 3. Prentice Hall, New Jersey (1996)
20. Lanotte, R., Tini, S.: Taylor approximation for hybrid systems. In: Morari, M., Thiele, L. (eds.) HSCC 2005. LNCS, vol. 3414, pp. 402–416. Springer, Heidelberg (2005). doi:10.1007/978-3-540-31954-2_26
21. Lee, E.A.: What's ahead for embedded software? Computer **33**(9), 18–26 (2000)
22. Liu, J., Lv, J., Quan, Z., Zhan, N., Zhao, H., Zhou, C., Zou, L.: A calculus for hybrid CSP. In: Ueda, K. (ed.) APLAS 2010. LNCS, vol. 6461, pp. 1–15. Springer, Heidelberg (2010). doi:10.1007/978-3-642-17164-2_1
23. Majumdar, R., Zamani, M.: Approximately bisimilar symbolic models for digital control systems. In: Madhusudan, P., Seshia, S.A. (eds.) CAV 2012. LNCS, vol. 7358, pp. 362–377. Springer, Heidelberg (2012). doi:10.1007/978-3-642-31424-7_28
24. Platzer, A.: Differential-algebraic dynamic logic for differential-algebraic programs. J. Logic Comput. **20**(1), 309–352 (2010)
25. Platzer, A.: The complete proof theory of hybrid systems. In: LICS, pp. 541–550. IEEE (2012)
26. Pola, G., Girard, A., Tabuada, P.: Approximately bisimilar symbolic models for nonlinear control systems. Automatica **44**(10), 2508–2516 (2008)
27. Pola, G., Pepe, P., Di Benedetto, M.: Symbolic models for networks of discrete-time nonlinear control systems. In: ACC, pp. 1787–1792. IEEE (2014)
28. Selic, B., Gérard, S.: Modeling and Analysis of Real-Time and Embedded Systems with UML and MARTE: Developing Cyber-Physical Systems. Elsevier, Amsterdam (2013)
29. Sontag, E.D.: Mathematical Control Theory: Deterministic Finite Dimensional Systems, vol. 6. Springer, Heidelberg (2013)
30. Stoer, J., Bulirsch, R.: Introduction to Numerical Analysis, vol. 12. Springer, Heidelberg (2013)
31. Tiller, M.: Introduction to Physical Modeling with Modelica, vol. 615. Springer, Heidelberg (2012)

32. Tiwari, A.: Abstractions for hybrid systems. Formal Methods Syst. Des. **32**(1), 57–83 (2008)
33. Wang, S., Zhan, N., Guelev, D.: An assume/guarantee based compositional calculus for hybrid CSP. In: Agrawal, M., Cooper, S.B., Li, A. (eds.) TAMC 2012. LNCS, vol. 7287, pp. 72–83. Springer, Heidelberg (2012). doi:10.1007/978-3-642-29952-0_13
34. Yan, G., Jiao, L., Li, Y., Wang, S., Zhan, N.: Approximate Bisimulation and Discretization of Hybrid CSP. CoRR, abs/1609.00091, August 2016
35. Zhan, N., Wang, S., Zhao, H.: Formal modelling, analysis and verification of hybrid systems. In: Liu, Z., Woodcock, J., Zhu, H. (eds.) Unifying Theories of Programming and Formal Engineering Methods. LNCS, vol. 8050, pp. 207–281. Springer, Heidelberg (2013). doi:10.1007/978-3-642-39721-9_5
36. Chaochen, Z., Ji, W., Ravn, A.P.: A formal description of hybrid systems. In: Alur, R., Henzinger, T.A., Sontag, E.D. (eds.) HS 1995. LNCS, vol. 1066, pp. 511–530. Springer, Heidelberg (1996). doi:10.1007/BFb0020972