Abstraction-based control synthesis using partial information

W. A. Apaza-Perez, C. Combastel, I. Walukiewicz, A. Muscholl, A. Zolghadri

Abstraction-based control synthesis using partial information

W.A. Apaza-Perez[a], C. Combastel[a], I. Walukiewicz[b], A. Muscholl[b], A. Zolghadri[a]

[a] *Univ. Bordeaux, CNRS, IMS, UMR 5218, 33405 Talence, France*
[b] *Univ. Bordeaux, CNRS, LaBRI, UMR 5800, 33405 Talence, France*

Abstract

The problem studied in this paper is that of distributed controller design for interconnected systems using abstraction-based techniques. Controller synthesis for each subsystem uses local distributed sensor information from other subsystems. Such partial information in an abstraction will be characterized in terms of ranking functions which can be deemed as level sets of Lyapunov functions. An effective procedure is proposed for the computation of ranking functions in the case of reach and stay specifications. A step-by-step algorithmic procedure implementing the proposed approach is presented for controller synthesis based on partial information. A numerical example is provided to illustrate the implementation.

Keywords: Control system synthesis; persistency specifications; symbolic control; compositional method.

1. Introduction

1.1. Context

The basic problem addressed in this paper is that of distributed control for interconnected systems. Many engineered systems such as electrical grids and power networks, or mechatronic systems, are modular subsystems, i.e. interacting networks of smaller dynamically coupled components which are interconnected through their inputs and outputs, where the interaction between control software with physical processes are present. Symbolic control is a computational approach to controller synthesis for nonlinear systems (see e.g. [1, 2] and the references therein). The main concept of symbolic control is that of the symbolic model, also called discrete abstraction, which is a finite state/input approximation of a general continuous dynamical system. When a symbolic model is related to the original system by some formal behavioral relationship such as alternating simulation [1] approximate bisimulations [3, 4, 5] or feedback refinement [6], controllers designed for the symbolic model can be refined into controllers for the original system. This makes it possible to use automatic controller synthesis techniques for finite state dynamical systems to synthesize controllers for continuous systems (see e.g. [2]). Symbolic control can be applied to general classes of continuous systems with state and input constraints and subject to bounded disturbances. Also, it enables the automatic synthesis of controllers that are "correct-by-design" for various type of specifications such as safety, reachability, attractivity or more complex properties such as those described in Linear Temporal Logic [2] that go beyond the traditional specifications (stabilization, output tracking or disturbance rejection) in control theory. Symbolic methods have spurred on substantial research efforts over the last two decades, these are often focused on different type of dynamics or different techniques for behavioral relationships between abstracted and original systems: dealing with the nondeterminism dynamic [7], non linear systems [8, 9], switched systems [10], stochastic systems [11], interconnected systems [12, 13, 14].

Note that a critical issue with existing symbolic control techniques is that they do not scale well, [15]. Some recent research works dealing with this issue are [16, 17, 18]. For example, [16] provides multiscale symbolic models that describe transitions by a sequence of embedded lattices approximating the state-space for incrementally stable switched systems, [17] proposes an optimization criterion for state-space grids based on a prediction of the computational effort for the abstraction to be computed for non linear systems. In [18], a compositional construction based in a small gain type condition for approximate abstractions of

interconnected control systems. Software tools are now available for the computation of abstractions, for example, PESSOA [19], CoSyMa [20], TuLiP [21], or SCOTS [22].

Our paper extends the work of [13] on small gain results and distributed control to an algorithmic approach for controlling arbitrary many interconnected components with respect to reach-and-stay objectives.

1.2. Related work

Several methods to design distributed control laws already exist in the literature (see [23] and the references therein). For interconnected systems, the compositional approach is very appealing for achieving tractable distributed control solutions which satisfy global system specifications, [18]. A similar problem occurs in formal verification where implementations of computer programs are checked for correctness. Here, compositional and assume-guarantee reasoning provides strategies to decompose a verification task, [24]. The notions of compositional abstractions are reported in some recent papers, usually to extend the range of physical systems that can be addressed. Among others, the interested reader can refer to [25, 26, 14, 18, 27] and the references therein. The paper [25] presents a compositional approach for approximate abstraction which performs a model order reduction from one continuous system to another continuous system with fewer state variables. In [26, 14] and [18], a compositional abstraction technique is proposed for networked continuous systems based on approximate bisimulation. The paper [27] introduces the so called disturbance bisimulation, as the basis for compositional symbolic abstractions.

1.3. Contributions

The developments offered in this paper have initially been motivated by the work reported in [13] that made use of small gain results from control theory and assume-guarantee reasoning from formal methods in two interconnected systems. The extension from two to n interconnected systems is not immediate due to the lack of explicit proofs, constructive algorithms and illustrative examples. These aspects were introduced and analyzed in a preliminary work reported in the conference communication [28] but without providing a complete solution yet.

As in the work of [13], this work considers the problem of enforcing a persistency specification of the form "reach a set of states P and remain there for all future time", which is denoted in Linear Temporal Logic (LTL) by $\Diamond\Box P$, meaning "eventually always", see [2]. The main contributions of this paper are: Firstly, the key feature of the proposed approach is that controller synthesis is based on local distributed sensor information from other subsystems. Such partial information is characterized in terms of Lyapunov-like ranking functions as introduced for the first time in [13], who also provided an algorithm to find the minimum cardinality of ranking function codomains in the case of $n = 2$ interconnected systems. Unfortunately, in many cases, codomains of minimal cardinality will not be sufficient to construct a controller enforcing the required specifications. Secondly, beyond the results in [13] and, to the best of our knowledge, beyond subsequent works, this paper develops a generic algorithm for constructing such ranking functions and a distributed controller provenly enforcing the satisfaction of a persistency specification by the abstracted interconnected system. The algorithm uses an original well-order to that purpose. Thirdly, the generic procedure is applied to a numerical example for which the method from [13] cannot be applied directly. Finally, those three contributions make it possible to explicitly compute lower and upper bounds for the minimum cardinality of ranking function codomains. In the proposed framework, the inequalities related to these lower (resp. upper) bounds correspond to necessary (resp. sufficient) conditions making it possible to enforce the local reach and stay specifications.

The paper is organized as follows. Some preliminary definitions are given in Section 2. Section 3 is devoted to the problem statement. In Section 4, a procedure based on Lyapunov-like (or ranking) functions for building a reduced discrete abstraction of the original interconnected system is proposed. In Section 5, an algorithm is proposed to explicitly compute those ranking functions and ensure that the persistency specification is satisfied. Section 6 presents a numerical example and some concluding remarks are given in Section 7.

2. Preliminaries

2.1. Notation.

The cardinality of a set A is denoted by $|A|$. The relative complement of the set A in the set B is denoted by $B \setminus A$. Given a relation $R \subseteq A \times B$ and $A_0 \subseteq A$, we define $R(A_0) = \{b \in B \,|\, \exists\, a \in A_0, (a, b) \in R\}$. $f : A \to B$ denotes an ordinary map, and $f^{-1}(b) := \{a \in A : f(a) = b\}$ for $b \in B$. The symbols $\mathbb{R}$, $\mathbb{Z}$, $\mathbb{N}_0$ denote the set of real numbers, integers, natural numbers including the zero, respectively. $[a; b]$ denotes a discrete interval with a and b as lower and upper bound, respectively. Given numbers $i, n \in \mathbb{N}$ with $i \leq n$, the following sets are defined where the element positions are given in ascending order with respect to their values; $\mathcal{I} = [1; n] \subset \mathbb{N}$, $\tilde{\mathcal{I}}^i = \mathcal{I} \setminus \{i\}$. Given two sets A and B, the product $A \times B$ denotes the Cartesian product, and for a collection of sets $\{A_j\}_{j \in \tilde{\mathcal{I}}^i}$, indexed by the set $\tilde{\mathcal{I}}^i$, the product $\prod_{j \in \tilde{\mathcal{I}}^i} A_j$ denotes the Cartesian product keeping the order in $\tilde{\mathcal{I}}^i$, i.e. $\prod_{j \in \tilde{\mathcal{I}}^i} A_j := A_1 \times \cdots \times A_{i-1} \times A_{i+1} \times \cdots \times A_n$. Given a vector $x = (x_1, ..., x_n) \in \mathbb{R}^n$, x_i denotes the i-th component of x, and $\tilde{x}_i$ is defined as $\tilde{x}_i = (x_1, ..., x_{i-1}, x_{i+1}, ..., x_n)$.

2.2. Transition systems.

Abstractions are dynamical systems with finitely many states and input values, each of which symbolizes aggregates of states and inputs of the original system. Abstractions are mathematically modeled as transition systems [1].

Definition 1 (Transition system). *A transition system S is a tuple (X_S, U_S, F_S, Y, H), where X_S is a set of states, U_S is a set of control inputs, $F_S \subseteq X_S \times U_S \times X_S$ is a transition relation, Y is a set of outputs, and $H : X_S \to Y$ is an output map. When $Y = X_S$ and the output map H is the identity function, then the transition system is reduced to a tuple $S = (X_S, U_S, F_S)$.*

Relation F_S says when a transition can occur from state x to state x' upon control input u. A transition $(x, u, x') \in F_S$ is also denoted by $x \xrightarrow{u} x'$. The notation $Post_u(x) = \{x' \in X_S : (x, u, x') \in F_S\}$ denotes the set of successors of x upon control input u.

Definition 2 (Alternating simulation relation). *Given two transition systems $S_a = (X_a, U_a, \ F_a)$ and $S_b = (X_b, U_b, F_b)$, a relation $R \subseteq X_a \times X_b$ is an alternating simulation relation from S_a to S_b if the following condition is satisfied:*

$$\forall (x_a, x_b) \in R,\ \forall u_a \in U_a,\ \exists u_b \in U_b,\ \forall x'_b \in Post_{u_b}(x_b), \exists x'_a \in Post_{u_a}(x_a),\ (x'_a, x'_b) \in R.$$

Alternating simulation relation allows the designer to work with the abstract system S_a instead of the concrete system S_b. For example, in the case of a reach-and-stay specification, if there is a suitable controller in S_a then there is one in S_b provided every state in S_b has a successor, i.e. $\forall x_b \in S_b$, $Post_{u_b}(x_b) \neq \emptyset$. Note also that the abstraction should not be too coarse: the states to reach-and-stay, as given by a specification, should be separated from other states in the abstraction. Under this condition, a controller for the abstract system S_a can be refined to a controller on the concrete system S_b, see [1].

The Feedback Refinement Relation as in [6] is a special case of alternating simulation relations:

- $U_a \subseteq U_b$;
- $\forall (x_a, x_b) \in R, \forall u \in U_a, R(Post_u(x_b)) \subseteq Post_u(x_a)$.

It ensures that when the abstract controller is applied to the original system, the temporal logic specifications are satisfied.

A dynamical system evolving in continuous-time can be modeled as a transition based on a τ-sampled behavior, $\tau > 0$, see e.g. [1]. We refer to [1] for other basic notions and the relation between bisimulation and the control synthesis problem. The interested reader can also refer to [4] for the notion of approximate simulation and bisimulation relations and functions, where the authors provide upper-bounds on the approximation metrics. Approximate bisimulation is a symmetric version of alternating simulation. The symbolic model can be constructed without stability assumptions [29]. It is shown in [30] that bisimulation unifies the concepts of state-space equivalence and state-space reduction. Moreover, the notion of bisimulation relation for general linear differential-algebraic systems is formulated in [31].

3. Problem statement

Consider a discrete interconnected system:

$$\begin{aligned} x_1^+ &\in f_1(x_1, x_2, x_3, ...x_n, u_1) \\ x_2^+ &\in f_2(x_2, x_1, x_3, ...x_n, u_2) \\ &\vdots \\ x_n^+ &\in f_n(x_n, x_1, x_2, ...x_{n-1}, u_n) \end{aligned} \tag{1}$$

with $x_i \in X_i$ and $u_i \in U_i$ for some finite sets X_i, U_i, $i \in \mathcal{I}$. The system (1) is discrete, however, it encompasses sampled versions and abstractions of continuous-time systems, possibly subject to disturbances, see [6, 32]. The system (1) is non-deterministic in the sense that if an input is applied in a state, several next states are possible. The trajectories of the system (1) are denoted by $\forall i \in \mathcal{I}$, $x_i(k, x_0, c_i)$ with initial condition $x_i(0, x_0, c_i) = x_0$, discrete time $k \in \mathbb{N}_0$, and control $c_i : X_i \times \prod_{j \in \tilde{\jmath}^i} X_j \to U_i$.

Consider specifications of the form "reach P and stay there", where $P = P_1 \times P_2 \times \cdots \times P_n$ for some sets $P_i \subseteq X_i$, using linear temporal logic notations these are written as $\Diamond\Box P$. The control problem is to find controllers $c_i : X_i \times \prod_{j \in \tilde{\jmath}^i} X_j \to U_i$ for each $i \in \mathcal{I}$, such that the system described by (1), under the state feedback controls $u_i = c_i(x_i, \tilde{x}_i)$, satisfies

$$\Diamond\Box P : \forall x_0 \in X, \exists\, k_i \in \mathbb{N}_0, \forall k \geq k_i, \;\; x_i(k, x_0, c_i) \in P_i \tag{2}$$

Our objective is to design a controller c_i for the subsystem i in a domain $X_i \times \prod_{j \in \tilde{\jmath}^i} D_j \times U_i$ of smaller cardinality than $X_i \times \prod_{j \in \tilde{\jmath}^i} X_j \times U_i$ by using a *reduced knowledge* about other subsystems. Our approach is based on ranking functions that characterize partial information about the sensed states of other subsystems. Alternating simulation relation will be used to infer the existence of a controller for (1) from the existence of a controller for a reduced discrete abstraction.

4. Construction of reduced discrete abstractions

The construction of reduced discrete abstractions are based mainly on controllable predecessors $CP_i\,(U_i, E, S_i)$, where U_i is the input set, $E \subseteq \prod_{j \in \tilde{\jmath}^i} X_j$ and $S_i \subseteq X_i$, as

$$CP_i\,(U_i, E, S_i) = \left\{ \;\; x_i \in X_i : \exists\, u_i \in U_i, \forall \tilde{x}_i \in E, f_i(x_i, \tilde{x}_i, u_i) \subseteq S_i \;\; \right\}, \tag{3}$$

which will be particularized using similar notations for specific purposes later.

The controllable predecessor (3) describes the states in X_i for which the controlled system i is able to reach the target set S_i despite the local influences, expressed by E, from other interconnected systems (robustness property). E may be inferred on some available partial knowledge about states of other components.

The transition system modeling the system (1) is denoted $S = (X_S, U_S, F_S)$ where $X_S = \prod_{i \in \mathcal{I}} X_i$, $U_S = \prod_{i \in \mathcal{I}} U_i$, and F_S is given by

$$F_S = \left\{ \;\; (x, u, x') \in X_S \times U_S \times X_S : \;\; \forall\, i \in \mathcal{I}, \;\; x_i' \in f_i(x_i, \tilde{x}_i, u_i) \;\; \right\}. \tag{4}$$

The construction of a reduced discrete abstraction T based on ranking functions is done as follows. Consider ranking functions defined in each subsystem $i \in \mathcal{I}$ by $V_i : X_i \to D_i$, where $D_i = \{0, 1, 2, ..., d_i\}$, for some $d_i \in \mathbb{N}_0$, and $|D_i| \leq |X_i|$; and given a subset $Z \subset X_i$ define V_i^M as

$$V_i^M(Z) = \max\{V_i(z) \,|\, z \in Z\}. \tag{5}$$

Intuitively, the ranking functions represent some notion of distance to the set $V_i^{-1}(0) \subseteq X_i$. In principle, they can be freely proposed, but we will give a constructive way to build them when considering persistency specifications in the next section.

To start with, adapt the functions f_i of (1) from the domain $X_i \times \prod_{j\in\tilde{\jmath}^i} X_j \times U_i$ to the domain $X_i \times \prod_{j\in\tilde{\jmath}^i} D_j \times U_i$ as follows

$$\forall i \in \mathcal{I},\ F_i(x_i, \tilde{v}_i, u_i) = \bigcup_{\tilde{x}_i \in \prod_{j\in\tilde{\jmath}^i} V_j^{-1}(v_j)} f_i(x_i, \tilde{x}_i, u_i). \tag{6}$$

The controllable predecessor based on values of ranking functions for each subsystem $i \in \mathcal{I}$ from a set $S_i \subseteq X_i$ under the influence of $\tilde{v}_i \in \prod_{j\in\tilde{\jmath}^i} D_j$ is defined as

$$CPre_i^{\tilde{v}_i}(S_i|U_i) = \left\{\ x_i \in X_i : \exists\, u_i \in U_i,\ F_i(x_i, \tilde{v}_i, u_i) \subseteq S_i\ \right\}. \tag{7}$$

Intuitively, $\tilde{v}_i$ is a value characterizing some partial information about the states of components other than i. The definition of predecessor (7) relies on the ranking functions $\{V_i\}_{i\in\mathcal{I}}$ and its relation with the generic definition (3) is given by (8):

$$CPre_i^{\tilde{v}_i}(S_i|U_i) = CP_i\left(U_i, \prod_{j\in\tilde{\jmath}^i} V_j^{-1}(v_j), S_i\right). \tag{8}$$

To simplify the notations, $V_i^{-1}(\leq v_i)$ will denote a shorthand for $\bigcup_{k\leq v_i} V_i^{-1}(k)$ with $k \in \mathbb{N}_0$. Consider the function $\mathbf{V}_i^+ : D_i \times \prod_{j\in\tilde{\jmath}^i} D_j \to D_i$ defined with the controllable predecessor:

$$\forall i \in \mathcal{I},\quad \mathbf{V}_i^+(v_i, \tilde{v}_i) = \min\left\{\ k \in \mathbb{N}_0 : V_i^{-1}(v_i) \subseteq CPre_i^{\tilde{v}_i}\left(V_i^{-1}(\leq k)\big|\, U_i\right)\ \right\}. \tag{9}$$

For such $\mathbf{V}_i^+$, consider the abstraction T given by $T = (X_T, U_T, F_T)$, where $X_T = \prod_{i\in\mathcal{I}} D_i$, $U_T = \{u_T\}$ for some control input u_T, and

$$F_T = \left\{\ (v, u_T, v') \in X_T \times \{u_T\} \times X_T : \forall\, i \in \mathcal{I},\ \ v_i' \leq \mathbf{V}_i^+(v_i, \tilde{v}_i)\ \right\}. \tag{10}$$

Lemma 1. *The relation $R \subseteq X_T \times X_S$ given by*

$$R = \left\{\ (v, x) \in X_T \times X_S : \forall\, i \in \mathcal{I},\ v_i = V_i(x_i)\ \right\}, \tag{11}$$

is an alternating simulation relation from T to S.

Proof. We show that the condition of Definition 2 is satisfied. For all $(v, x) \in R$, the equality

$$\forall\, i \in \mathcal{I},\quad v_i = V(x_i), \tag{12}$$

is satisfied. Define $\mathbf{V}_i^+(v_i, \tilde{v}_i)$ as in (9), which implies

$$\forall\, i \in \mathcal{I},\quad V_i^{-1}(v_i) \subseteq CPre_i^{\tilde{v}_i}\left(V_i^{-1}\left(\leq \mathbf{V}^+(v_i, \tilde{v}_i)\right)\big|\, U_i\right). \tag{13}$$

From (12), $\forall i \in \mathcal{I}$, $x_i \in V_i^{-1}(v_i)$ is ensured. By (13) we get $x_i \in CPre_i^{\tilde{v}_i}\left(V_i^{-1}\left(\leq \mathbf{V}^+(v_i, \tilde{v}_i)\right)\big|\, U_i\right)$. The controllable predecessor $CPre$ in (7) implies the existence of a control $u \in U_S$ such that

$$\forall\, i \in \mathcal{I},\ \ F_i(x_i, \tilde{v}_i, u_i) \subseteq V_i^{-1}\left(\leq \mathbf{V}_i^+(v_i, \tilde{v}_i)\right). \tag{14}$$

Through equations (1) and (6), one gets that $\forall x' \in Post_u(x_i, \tilde{x}_i)$; meaning $x_i' \in F_i(x_i, \tilde{v}_i, u_i)$ for each $i \in \mathcal{I}$. By inclusion (14) one obtains

$$\forall\, i \in \mathcal{I},\quad x_i' \in \bigcup_{k\leq \mathbf{V}_i^+(v_i, \tilde{v}_i)} V_i^{-1}(k), \tag{15}$$

which implies the existence $v' \in Post_{u_T}(v)$ such that $0 \leq v_i' \leq \mathbf{V}_i^+(v_i, \tilde{v}_i)$ and $x_i' \in V_i^{-1}(v_i')$, i.e. $V_i(x_i') = v_i'$ for each $i \in \mathcal{I}$. Consequently $(v', x') \in R$ by (11). □

The next Theorem 1 extends Theorem 3.4 in [13] from two to n interconnected systems: it gives the domain of admissible controllers for T satisfying the persistency specification.

Theorem 1. *Suppose that T satisfies the specification $\Diamond\Box P_T$, for some set $P_T \subseteq X_T$. Then there exists a controller $c = (c_1, ..., c_n)$, where c_i has domain $X_i \times D_1 \times \cdots \times D_{i-1} \times D_{i+1} \times \cdots \times D_n$ enforcing the specification $\Diamond\Box P_S$ given by*

$$P_S = \{x \in X : (V_1(x_1), V_2(x_2), ..., V_n(x_n)) \in P_T\}. \tag{16}$$

In this case, the controller can be chosen as follows

$$c_i(x_i, \tilde{v}_i) \in \left\{ u_i \in U_i : \max_{\tilde{x}_i \in \prod_{j\in\tilde{\jmath}^i} V_j^{-1}(v_j)} V_i^M(f_i(x_i, \tilde{x}_i, u_i)) \leq \mathbf{V}_i^+(V_i(x_i), \tilde{v}_i) \right\} \tag{17}$$

Proof. First we show that the condition in (17) is well defined. Let $(x_i, \tilde{v}_i) \in X_i \times \prod_{j\in\tilde{\jmath}^i} D_j$ and define $\mathbf{V}_i^+(V_i(x_i), \tilde{v}_i)$ as in (9). The inclusion (18) results from (9) by considering the minimal value of k, i.e. $k = \mathbf{V}_i^+(v_i, \tilde{v}_i)$, and $v_i = V(x_i)$:

$$V_i^{-1}(V_i(x_i)) \subseteq CPre_i^{\tilde{v}_i}\left(V_i^{-1}\left(\leq \mathbf{V}_i^+(V_i(x_i), \tilde{v}_i)\right) \middle| U_i\right). \tag{18}$$

Note that $x_i \in V_i^{-1}(V_i(x_i))$ and consequently

$$x_i \in CPre_i^{\tilde{v}_i}\left(V_i^{-1}\left(\leq \mathbf{V}_i^+(V_i(x_i), \tilde{v}_i)\right) \middle| U_i\right).$$

From (7), $\exists u_i \in U_i$, $F_i(x_i, \tilde{v}_i, u_i) \subseteq V_i^{-1}(\leq \mathbf{V}_i^+(V(x_i), \tilde{v}_i))$, i.e. $\exists u_i \in U_i$, $V_i^M(F_i(x_i, \tilde{v}_i, u_i)) \leq \mathbf{V}_i^+(V(x_i), \tilde{v}_i)$. From (6), $\exists u_i \in U_i$, $\forall \tilde{x}_i \in \prod_{j\in\tilde{\jmath}^i} V_j^{-1}(v_j)$, $V_i^M(f_i(x_i, \tilde{x}_i, u_i)) \leq \mathbf{V}_i^+(V_i(x_i), \tilde{v}_i)$.

Now, consider a sequence

$$x \xrightarrow{u(1)} x(1) \xrightarrow{u(2)} \cdots \xrightarrow{u(k)} x(k) \xrightarrow{u(k+1)} x(k+1) \cdots \tag{19}$$

using controllers defined as in (17) for X_S, take the sequence

$$v \xrightarrow{u_T} v(1) \xrightarrow{u_T} \cdots \xrightarrow{u_T} v(k) \xrightarrow{u_T} v(k+1) \cdots \tag{20}$$

defined from (19) by $v_i(k) = V_i(x_i(k))$ for $i \in \mathcal{I}$ and $k \in \mathbb{N}$. By the assumption, T satisfies the specification $\Diamond\Box P_T$, then there exists $N \in \mathbb{N}$ such that $v(k) \in P_T$ for all $k \geq N$. As a consequence of Lemma 1, $x(k) \in P_S$ for $k \geq N$. □

Theorem 1 gives an explicit admissible set of controllers only when the system T satisfies $\Diamond\Box P_T$. This property can be checked by analyzing cycles in T.

Definition 3. *A directed graph $G = (\mathcal{V}, \mathcal{E})$ consists of a vertex set $\mathcal{V}$ and an edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. A cycle is a sequence of vertices $c(1), c(2), ..., c(m) \in \mathcal{V}$ such that $c(1) = c(m)$ and $(c(i), c(i+1)) \in \mathcal{E}$ for all i. A cycle is called self-cycle when $m = 2$, i.e. $(c(1), c(1)) \in \mathcal{E}$.*

The system T defined by (9)-(10) can be considered as a directed graph $G = (X_T, \mathcal{E}_T)$, where $(v, v') \in \mathcal{E}_T$ if and only if $(v, u_T, v') \in F_T$ for the unique u_T. A necessary and sufficient condition under which system T enforces the specification $\Diamond\Box P_T$, for some set $P_T \subseteq X_T$, can be obtained in terms of cycle properties:

Proposition 1. *Consider T defined in (9)-(10), and a target set $P_T \subseteq X_T$. T satisfies the specification $\Diamond\Box P_T$ if and only if all cycles in $G = (X_T, \mathcal{E}_T)$ are included in P_T.*

Proof. *Necessary condition:* assume that T satisfies $\Diamond\Box P_T$ and consider a cycle $v(1), v(2), ..., v(m) = v(1)$ that is reachable from some initial state $w(1)$, through a path $w(1) \to w(2) \to ... \to w(k) = v(1)$. Since the infinite path $w(1) \to w(2) \to ... \to v(1) \to ... \to v(m-1) \to ...$ reaches P_T and stays there forever, all vertices of the cycle must belong to P_T.

Sufficient condition, this is proved by contradiction: assume that all reachable cycles are included in P_T, and consider an infinite path from some initial state. If this path does not satisfy $\Diamond\Box P_T$ then some vertex

that appears infinitely often on this path does not belong to P_T. But this means that some reachable cycle is not included in P_T, a contradiction. □

The following proposition gives sufficient conditions under which the system T defined by (9)-(10) satisfies $\Diamond\Box P_T$ specification. Recall that the domain of T is a set of tuples of natural numbers, $X_T = \prod_{i\in\mathcal{J}} D_i$. The proposition uses a particular ordering on these tuples.
We write $v \sqsubset w$ if either:

1. the maximal value in v is smaller than that in w, $\max\{v_i : i = 1,\dots,n\} < \max\{w_i : i = 1,\dots,n\}$; or
2. the maximal values are the same, say k, and the last occurrence of k in v is before that in w, $\max\{i : v_i = k\} < \max\{j : w_j = k\}$.

Observe that $\sqsubset$ is a strict well-order on X_T ($\sqsubset$ is transitive and anti-symmetric), namely, there is no infinite sequence $v_1, v_2, \dots$ with $v_{i+1} \sqsubset v_i$, for $i = 1, \dots$

Proposition 2. *Take a system $T = (X_T, U_T, F_T)$ defined by (9)-(10). Let $P_T \subseteq X_T$ be a $\sqsubset$-downward closed set, i.e. a set such when $v \in P_T$ and $w \sqsubset v$ then $w \in P_T$. Consider two properties:*

i) $(\boldsymbol{V}_1^+(v_1,\tilde{v}_1),\dots,\boldsymbol{V}_n^+(v_n,\tilde{v}_n)) \sqsubset v$, *for all* $v \in X_T \setminus P_T$;
ii) $(\boldsymbol{V}_1^+(v_1,\tilde{v}_1),\dots,\boldsymbol{V}_n^+(v_n,\tilde{v}_n)) \in P_T$, *for all* $v \in P_T$.

System T satisfies the specification $\Diamond\Box P_T$ if conditions i), ii) are satisfied.

Proof. From the definition (10) of the transition relation in T, we get that if $v \to w$ then $w(i) \le \mathbf{V}_i^+(v_i,\tilde{v}_i)$ for all $i = 1,\dots,n$. Hence, by property i), $w \sqsubset v$. As we have remarked above, $\sqsubset$ is a well order so every sequence $v_1 \to v_2 \to \cdots$ must eventually reach a state from P_T. This shows that T satisfies $\Diamond P_T$.

Using the same observation, since P_T is $\sqsubset$-downwards closed, from property ii) we deduce that if $v \in P_T$ and $v \to w$ then $w \in P_T$. So T satisfies $\Diamond\Box P_T$. □

In this section, we have shown that it is possible to build reduced discrete abstractions from the ranking functions. Moreover, these can provide an explicit admissible set of controllers provided some conditions on the transition relation are satisfied. In the next section, the problem of constructing ranking functions ensuring that the reduced system satisfies these conditions will be considered.

5. Ranking functions and persistency specifications

An important result in [13] is to find a minimum cardinality of the codomain of the ranking functions, for the dimension $n = 2$. The first step in this section is to extend this result to multiple systems. Unfortunately, in many cases the resulting domains of minimal cardinality will not be sufficient to construct a desired controller. Namely, the reduced system T may not satisfy reach-and-stay specification. In order to tackle these restrictions, the first step in this section consists in extending the minimum cardinality result to an arbitrary number of subsystems ($n \ge 2$). Then, in the second subsection, an algorithm is presented for building ranking functions satisfying the (reach-and-stay) specifications that solves more control instances than [13].

5.1. Specifications and lower bound on codomain cardinality of the ranking functions

The lower bound on codomain cardinality of the ranking functions, such that the system T defined from (9)-(10) may satisfy the specification $\Diamond\Box P_T$, is found through the controllable predecessor parameterized by the input defined as in (21), which is an extension to multiple subsystems of the definition proposed in [13]:

$$CPre_i\left(S_i \middle| \prod_{j\in\tilde{\jmath}^i} X_j, U_i\right) = \left\{ \begin{array}{c} x_i \in X_i : \exists \tilde{x}_i \in \prod_{j\in\tilde{\jmath}^i} X_j, \\ \exists\, u_i \in U_i, f_i(x_i,\tilde{x}_i,u_i) \subseteq S_i \end{array} \right\}. \tag{21}$$

Note that the controllable predecessor definition given in (21) does not require information about ranking functions unlike the definition (8).

The controllable predecessor given in (21) can be used to define a sequence of sets $H_i^k \subseteq X_i$, $i \in \mathcal{I}$, $k \in \mathbb{N}_0$ as

$$\begin{aligned} H_i(0) &= P_i, \\ H_i(k+1) &= CPre_i\left(H_i(k) \left| \prod_{j\in\tilde{\jmath}^i} X_j, U_i\right.\right). \end{aligned} \tag{22}$$

The ranking functions $V_i^H : X_i \to \mathbb{N}_0$, $i \in \mathcal{I}$ defined from the sets $H_i(k)$'s are given by

$$V_i^H(x) = \min\{k \in \mathbb{N}_0 : x \in H_i(k)\}. \tag{23}$$

Note that the image of V_i^H, denoted by $D_i^{\min}$, has finite cardinality, due to the finite cardinality of X_i and U_i. The function V_i^H provides the minimum number of sequence of controls and states to reach the target set P_i by considering that the exact knowledge of the elements of the other subsystems X_j, $j \in \mathcal{I}^i$ would be required. Though useful to obtain the searched lower cardinality bounds, the functions V_i^H's are not necessarily good candidates for defining ranking functions that satisfy the conditions of Theorem 1, which motivates the search for techniques to define ranking functions as described in the next section 5.2.

In [13], a proposition about the minimum number of distinct values taken by the ranking functions (i.e. minimum cardinality of D_i) is given in the case of two subsystems. Its extension to n subsystems is straightforward. To avoid duplicating materials, the proof is omitted here.

Proposition 3. *Consider any function $V_i : X_i \to \mathbb{N}_0$ satisfying $V_i^{-1}(0) = P_i$ and whose image D_i has a cardinality strictly less than $|D_i^{\min}|$. For this V_i and arbitrary functions V_j, $j \neq i$, construct $\boldsymbol{V}_i^+$ as in (9), and construct the abstraction T in accordance with (10). Then T does not enforce the specification $\Diamond\Box\prod_{i\in\mathcal{I}}\{0\}$.*

One can define ranking functions such that $|D_i^{\min}| \leq |D_i|$, e.g. V_i^H in (23), but this is not sufficient to guarantee that the system T satisfies the specification $\Diamond\Box P_T$ since there is no guarantee that all cycles have vertices solely in P_T, see Proposition 1, is necessarily satisfied by any set of n ranking functions V_i with cardinality higher than $|D_i^{min}|$, $i \in \mathcal{I}$.

5.2. Ranking functions for satisfying specifications

An algorithm is presented for constructing distributed controllers that works in more cases than the procedure described in the previous subsection 5.1. It constructs a sequence of sets $Z_i(k)$, for $i \in \mathcal{I}$ and $k \in \mathbb{N}_0$. These sets will be used to define ranking functions. These in turn determine an abstract transition system T as in (9)-(10). Theorem 2 will show that T satisfies the specification $\Diamond\Box P_T$ for the target set P_T containing only the zero vector. Then, it will be possible to use Theorem 1 to obtain a distributed controller.

A rough idea behind Algorithm 1 is the following. The target sets P_i's are defined as initial sets of the algorithm in the step 1, where sets $Z_i(0)$'s guarantee that there are controllers enforcing the system to stay in target states. We assume that the system itself should satisfy stay specification in the target set, which implies the existence of controls enforcing it, step 1 identifies the sets $Z_i(0)$'s where the stay specification is satisfied, which are contained in the target sets P_i's. In step 2, for every i, we compute a set of states T_i for which there is a control input permitting to reach the set $Z_i(\leq k-1)$. The information available at this point is that each component $j \neq i$ is in a state from the set $Z_j(\ell_j)$. The *unsafe* condition in step 2 verifies that the new set $Z_i(k)$ does not influence unfavorably other components, namely there is a way to keep them in sets Z_j computed previously (see Fig. 1).

From the sets $Z_i(k)$ computed by the algorithm, ranking function for every $i \in \mathcal{I}$ can be defined as:

$$V_i(x) = \min\{k \in \mathbb{N}_0 : x \in Z_i(k)\}, \tag{24}$$

The domain of this n-tuple of functions is

$$CD = \prod_{i\in\mathcal{I}} Z_i(\leq k_{max}), \tag{25}$$

Algorithm 1: Sets for building Ranking Functions

Input: Target sets $P_i \subseteq X_i$, $i = 1, ..., n$;
Output: Sequence of sets $Z_i(k) \subseteq X_i$, for $i \in \mathcal{I}$, and $k \in \mathbb{N}_0$

```
// PART 1: Stay specification
```
1 $Z_1(0) \leftarrow P_1, ..., Z_i(0) \leftarrow P_i, ..., Z_n(0) \leftarrow P_n$;
2 **for** i=1,n **do**
3 $\quad Q \leftarrow \emptyset$;
4 $\quad$ **while** $Z_i(0) \not\subseteq Q$ **do**
5 $\quad\quad Q \leftarrow Z_i(0)$;
6 $\quad\quad Z_i(0) \leftarrow Q \cap CP_i(U_i, \prod_{s \in \tilde{\jmath}^i} Z_s(0), Q)$;

```
// PART 2: Reach specification
```
7 **while** $\exists i \in \mathcal{I}, Z_i(k) \neq \emptyset$ **do**
8 $\quad k \leftarrow k + 1$;
9 $\quad$ **for** $i \leftarrow 1, n$ **do**
10 $\quad\quad L(i) \leftarrow \{0, 1, ..., k\}^{(i-1)} \times \{k\} \times \{0, 1, ..., k-1\}^{(n-i)}$;
11 $\quad\quad T_i \leftarrow X_i$;
12 $\quad\quad$ **for** $\ell \in L(i)$ **do**
13 $\quad\quad\quad T_i \leftarrow T_i \cap CP_i\left(U_i, \prod_{s \in \tilde{\jmath}^i} Z_s(\ell_s), Z_i(\leq (k-1))\right)$; // ℓ_s is the s-th component of ℓ
14 $\quad\quad Z_i(k) \leftarrow T_i \setminus Z_i(\leq (k-1))$;
15 $\quad\quad$ *unsafe* $\leftarrow$ false;
16 $\quad\quad$ **for** $\ell \in L(i)$ ***and*** $j \in \{1, \ldots, n\} - \{i\}$ **do**
17 $\quad\quad\quad$ **if** $j < i$ **then**
18 $\quad\quad\quad\quad b \leftarrow k$ **else** $b \leftarrow (k-1)$;
19 $\quad\quad\quad$ **if** $Z_j(\leq b) \not\subseteq CP_j(U_j, \prod_{s \in \tilde{\jmath}^i} Z_s(\ell_s), Z_j(\leq b))$ **then**
20 $\quad\quad\quad\quad$ *unsafe* $\leftarrow$ true
21 $\quad\quad$ **if** *unsafe* **then**
22 $\quad\quad\quad Z_i(k) \leftarrow \emptyset$;

which coincides with the domain of the distributed controller, where k_{max} is the value of k at the end of the algorithm. Since the sets $Z_i(k)$ are pairwise disjoint, it comes:

$$V_i^{-1}(v_i) = Z_i(v_i). \tag{26}$$

Theorem 2. *Consider the system as in* (1)*, the sets $Z_i(k)$ computed by Algorithm 1, and ranking functions V_i as in* (24)*. Let T be the system defined according to* (9)-(10)*, and let $P_T = \{\mathbf{0}\}$ be a singleton set consisting of the zero vector. Then, system T satisfies the specification $\Diamond\Box P_T$.*

Proof. We use Proposition 2: it is enough to show that the two conditions of the proposition hold. These are treated in the two claims below.

Claim 1: For all $v \in X_T \setminus P_T$, $(\mathbf{V}_1^+(v_1, \tilde{v}_1), \ldots, (\mathbf{V}_n^+(v_n, \tilde{v}_n)) \sqsubset v$.

Let $v \in X_T \setminus P_T$, then $\exists i \in \mathcal{I}$ such that $v_i > 0$, define

$$k^* = \max\{v_s : s \in \mathcal{I}\}, \qquad i^* = \max\{s \in \mathcal{I} : v_s = k^*\}, \tag{27}$$

where $v_{i^*} = k^* > 0$ holds. From now on, consider the k^*-th iteration of Algorithm 1, line 10 ensures the next holds

$$\begin{cases} v_i \leq k^*, \text{ if } i \leq i^*; \\ v_i < k^*, \text{ if } i > i^*. \end{cases} \tag{28}$$

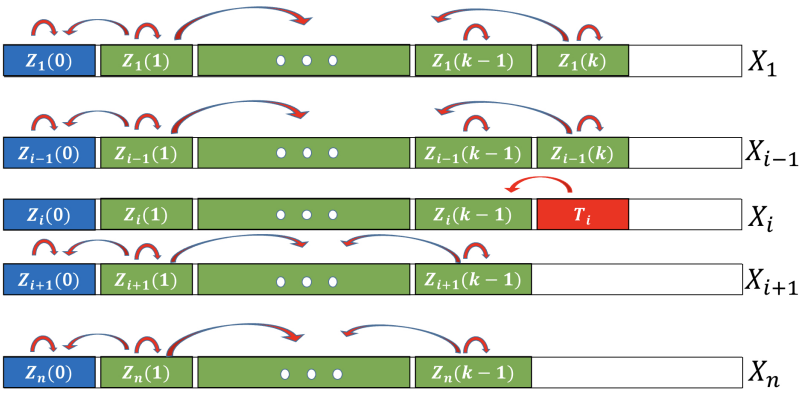

Figure 1: Description of the k-th iteration of Algorithm 1. The arrows illustrate the possible behaviour of Z's: i) the new set T_i reaches $Z_i(\leq k-1)$ despite the interactions with all other subsets previously defined as Z'_js, $j \in \tilde{\jmath}^i$, ii) the effect of the new set T_i in the Z_j's keeps them inside $Z_j(\leq k)$ for $j < i$ and $Z_j(\leq k-1)$ for $i < j$ (safe condition).

Consider in Algorithm 1 (lines 8-9) the parameters $k \leftarrow k^*$, $i \leftarrow i^*$ and $\ell \leftarrow v$ (line 10) to ensure

$$T_{i^*} \leftarrow T_{i^*} \cap CP_{i^*}(U_{i^*}, \prod_{s\in\tilde{\jmath}^{i^*}} Z_s(v_s), Z_{i^*}(\leq (k^*-1))),$$

which implies (by lines 13-14)

$$Z_{i^*}(k^*) \subseteq CP_{i^*}(U_{i^*}, \prod_{s\in\tilde{\jmath}^{i^*}} Z_s(v_s), Z_i(\leq (k^*-1))) \ .$$

From (26), we have $Z_i^*(k^*) = V_{i^*}^{-1}(k^*)$, then the above inclusion can then be rewritten as

$$V_{i^*}^{-1}(k^*) \subseteq CP_{i^*}(U_{i^*}, \prod_{s\in\tilde{\jmath}^{i^*}} V_s^{-1}(v_s), V_{i^*}^{-1}(\leq (k^*-1)))$$

From $\mathbf{V}^+$ definition given in (9), we have

$$\mathbf{V}_{i^*}^+(v_{i^*}, \tilde{v}_{i^*}) \leq k^* - 1 < k^* = v_{i^*}. \tag{29}$$

Now, let $j \in \tilde{\jmath}^{i^*}$ and $\ell \leftarrow v$ in Algorithm 1 (lines 16-20) which imply, in terms of ranking functions (26),

$$\begin{cases} V_j^{-1}(\leq k^*) \subseteq CP_j(U_j, \prod_{s\in\tilde{\jmath}^j} V_s^{-1}(v_s), V_j^{-1}(\leq k^*)) & \text{if } j < i^* \\ V_j^{-1}(\leq k^*-1) \subseteq CP_j(U_j, \prod_{s\in\tilde{\jmath}^j} V_s^{-1}(v_s), V_j^{-1}(\leq k^*-1)) & \text{if } j > i^* \end{cases} \tag{30}$$

where the variable *unsafe* must be false due to that $Z_{i^*}(k^*) \neq \emptyset$. From (22), (30) and V^+ definition (9), we obtain

$$\begin{cases} \mathbf{V}_j^+(v_j, \tilde{v}_j) \leq k^* & \text{if } j < i^* \\ \mathbf{V}_j^+(v_j, \tilde{v}_j) \leq k^* - 1 & \text{if } j \geq i^* \end{cases} \tag{31}$$

with $V_j^{-1}(v_j) \subseteq V^{-1}(\leq k^*)$ if $j < i^*$, and $V_j^{-1}(v_j) \subseteq V^{-1}(\leq k^*-1)$ if $j > i^*$. Therefore, the inequalities in (31) imply $(\mathbf{V}_1^+(v_1, \tilde{v}_1), \ldots, (\mathbf{V}_n^+(v_n, \tilde{v}_n)) \sqsubset v$ holds.

Claim 2: $\mathbf{V}_i^+(0, \mathbf{0}) = 0$ for all $i = 1, \ldots, n$.

This claim ensures that P_T is a $\sqsubset$-downward closed set and condition ii) in Proposition 2 is satisfied. The

inclusion $Z_i(0) \subseteq CP_i(U_i, \prod_{s\in\tilde{\jmath}^i} P_s(0), Z_i(0))$ holds by step 1 in Algorithm 1. This means in terms of ranking functions:

$$V_i^{-1}(0) \subseteq CP_i(U_i, \prod_{s\in\tilde{\jmath}^i} V_s^{-1}(0), V_i^{-1}(0)), \tag{32}$$

where for each $i = 1; n$ $Z_i(0) \subseteq P_i$ is satisfied. By (8), the above can be written as $V_i^{-1}(0) \subseteq CPre_i^{\tilde{\mathbf{0}}^i}\left(V_i^{-1}(0)|U_i\right)$. By definition of $\mathbf{V}_i^+$ in (9), we get $\mathbf{V}_i^+(0,\mathbf{0}) = 0$ as desired. □

Consequently, Theorem 2 provides a constructive way to satisfy the main assumptions of Theorem 1. Therefore, it should be emphasized that the online implementation of the designed subsystem controllers (17) indeed presents interesting features: instead of a unique centralized controller, significantly simpler subsystem controllers result from the proposed design. In particular, since only partial information (ranking function values) is used from other subsystems, the number of symbolic state configurations to explore in order to find an adequate control value is drastically reduced from $|X_i \times \prod_{j\in\tilde{\jmath}^i} X_j|$ to $|X_i \times \prod_{j\in\tilde{\jmath}^i} D_j|$.

6. Example: Floor Heating

In this section, the theoretical results of this paper are illustrated with the temperature regulation in a house with 3 rooms, two of them being equipped with a heater. This example is selected to illustrate the management of continuous dynamics in a symbolic framework. It is assumed that the exterior temperature remains constant during the simulations. For each room $i \in \{1, 2, 3\}$, the variations of the temperature T_i are described by the following system adapted from the model presented in [33]:

$$\begin{aligned}
\dot{T}_1(t) &= A_{1,2}(T_2(t) - T_1(t)) + B_1(T_{env}(t) - T_1(t)) + H_1(t)\\
\dot{T}_2(t) &= A_{2,1}(T_1(t) - T_2(t)) + A_{2,3}(T_3(t) - T_2(t)) + B_2(T_{env}(t) - T_2(t))\\
\dot{T}_3(t) &= A_{3,2}(T_2(t) - T_3(t)) + B_3(T_{env}(t) - T_3(t)) + H_3(t)
\end{aligned} \tag{33}$$

where $A_{i,j}$ is the heat exchange factor specific to walls and windows, T_{env} is the current outside temperature (considered as a disturbance), B_i is the heat exchange coefficient between outside temperature and room i, $H_1(t) \in \{0, h_{e_1}\}$ and $H_3(t) \in \{0, h_{e_3}\}$ are the power of the heaters in the rooms 1 and 3, respectively. The numerical values are $A_{1,2} = A_{2,1} = A_{2,3} = A_{3,2} = 0.004$, $B_1 = B_2 = B_3 = 0.003$, $h_{e_1} = 0.2$, $h_{e_3} = 0.3$, and $4^oC \leq T_{env} \leq 5^oC$.

The continuous-time dynamics of (33) is periodically sampled with period 12 seconds. We impose the state constraint $T_i \in [0, 30]$, $i = 1, \ldots, 3$ and the control objective is to stabilize the temperatures in the intervals: $T_1, T_3 \in [19, 23]$ and the allowed range of variation of T_2 is selected to be looser $T_2 \in [10, 23]$.

The approach described in [32] is used to compute symbolic abstractions. For that purpose, we use uniform partitions made of 60 sub-intervals for each of the three state intervals $[0, 30]$. Uniform discretizations of the input H_j with 2 elements $\{h_{e_i}\}$ for $j = 1, 3$ are also considered, thus resulting in 4 distinct input symbols in the obtained abstraction.

The algorithm 1 is then used to synthesize ranking functions for the system. The algorithm stops after 12 iterations of the main loop. The total computation time is 297 seconds (CPU: 2.2 GHz Intel Core i7, RAM: 16 Go 1600 MHz DDR3, Matlab R2019b), with 114 seconds spent on computing the abstraction, 62 seconds spent in defining the ranking functions iteratively, and 121 seconds spent on control synthesis. In the computation of sets $Z_i(k)$'s, we obtain the values $d_1 = 11$, $d_2 = 12$, $d_3 = 11$ corresponding to the maximum values obtained for the codomain of the ranking functions. The state space can be decomposed using level sets of ranking functions where the objective is to decrease in the level sets until reaching the target set, see Figure 2. A better illustration of this decomposition of the state space can be obtained through a projection in two dimensions, see Figure 3.

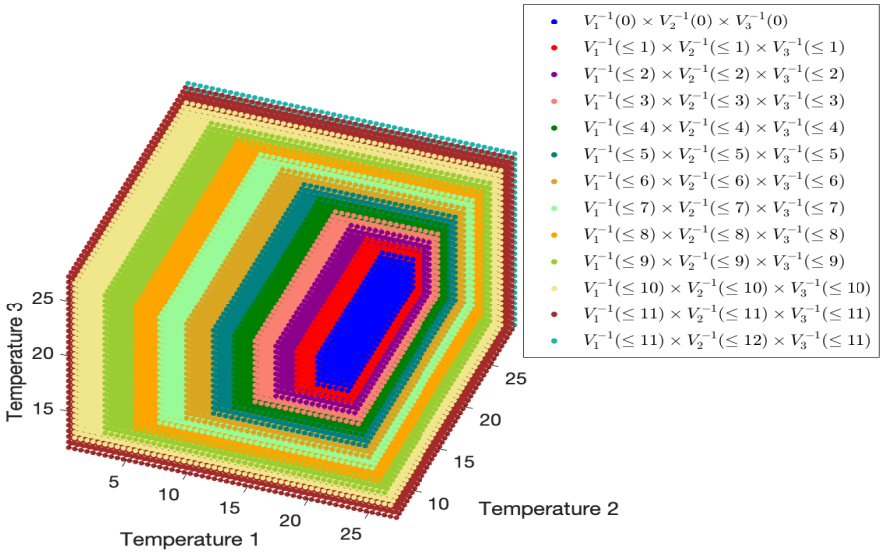

Figure 2: Level sets corresponding to the ranking functions, where the blue set in the center is the target set.

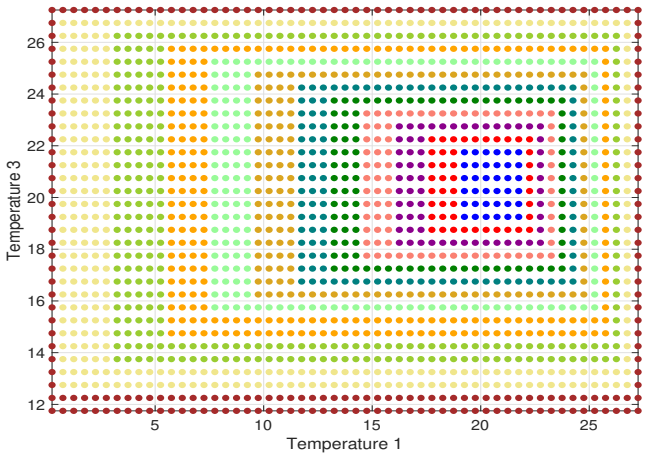

Figure 3: A two-dimensional projection with $T_2 = 20^oC$ from level sets corresponding to the ranking functions.

The proposed approach constructively builds ranking functions satisfying the acyclic property described in Proposition 1 (and thus the reach and stay specification), see Fig. 4. Blue arrows represent the transition relation of the resulting reduced discrete abstraction and red points highlight the states for which self-cycles may occur. As expected to satisfy the reach and stay specification, no self-cycles occur except for $(0, 0, 0)$ which represents the target set. By comparison, the direct extension to multiple interconnected subsystems of the ranking functions defined in [13] to derive the minimum cardinality result (Proposition 3) leads to a reduced discrete abstraction that does not satisfy the required acyclic property (Proposition 3.5 in [13] and Proposition 1 in this paper). It is thus a contribution of this paper to propose an algorithm constructively building Lyapunov-like discrete ranking functions satisfying reach and stay specifications in an interconnected framework.

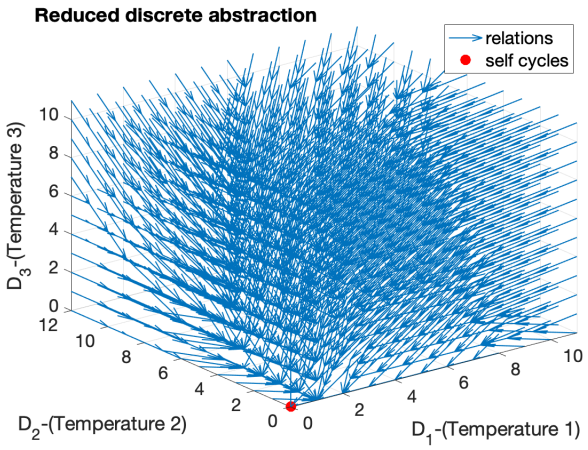

Figure 4: Reduced discrete abstraction resulting from Algorithm 1, it features no self-cycles except for $(0, 0, 0)$ which represents the target set.

.

Moreover, by combining the results of paragraphs 5.1 and 5.2, both a lower and an upper bound for the minimum cardinality of ranking function codomains enforcing the reach and stay specification of n interconnected subsystems is obtained. In the case of the numerical example developped in this section, this gives[1] the following integer intervals : $d_1 \in [7; 11] \cap \mathbb{N}$, $d_2 \in [8; 12] \cap \mathbb{N}$, $d_3 \in [6; 11] \cap \mathbb{N}$. In the proposed framework, the inequalities related to these lower (resp. upper) bounds correspond to necessary (resp. sufficient) conditions on the minimum cardinality of ranking function codomains making it possible to satisfy the reach and stay specification.

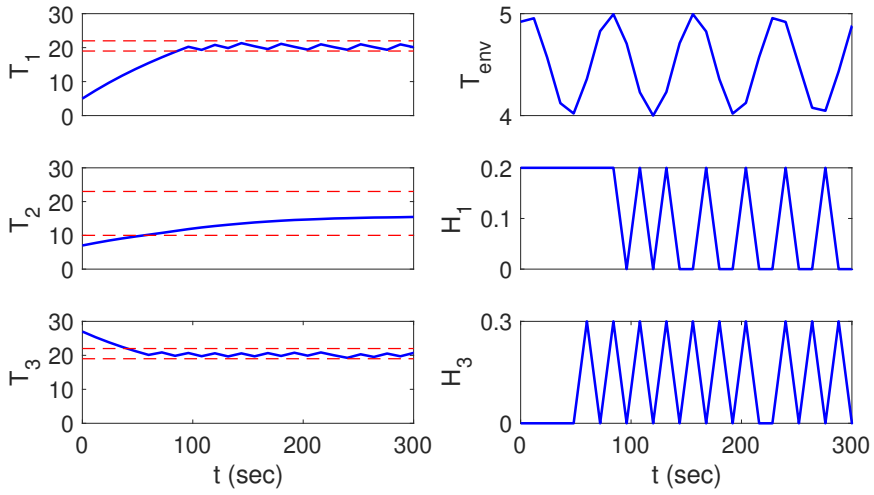

Figure 5: Simulated trajectories of system (33): (left side) evolution of the temperatures in each room, where the limits of the target region are represented by dashed lines; (right side) the outside temperature (considered as a disturbance) and the control inputs of room 2 and 3.

The implementation of the symbolic controller for system (33) is done through the controllers C_i defined in Theorem 1. Figure 5 shows a simulation of system (33) in the following scenario: the initial values of

[1]Notice that the cardinality of the codomain D_i of the ranking function V_i is indeed $d_i + 1$ since $D_i = \{0, 1, \ldots, d_i\} \subset \mathbb{N}$ also contains 0.

(T_1, T_2, T_3) are $(5, 7, 27)$. The outside temperature T_{env} is considered as a disturbance: for the numerical simulation, $T_{env}(t) = (\sin(t) + 9)/2$. The plots represent the temporal evolution of the temperatures in each room, the outside temperature and the control inputs. The trajectories reach the target set (limited by dashed lines) after 100 seconds and stay in that region afterwards. Most of the computational effort is done off-line at the design step. Indeed, the numerical simulation of the "on-line" closed-loop scenario is fast, around 3 ms per sample for the example reported in Figure 5 and, as expected, the controllers resulting from the symbolic abstractions fulfill the specifications when applied to the concrete system.

7. Conclusions

The symbolic design of a distributed controller scheme enforcing persistency specification for n interconnected non-deterministic systems is addressed in this work. The resulting local controllers do not require the knowledge of the full state as they are based on ranking functions characterizing partial information used in the considered abstraction. This results in lower complexity controllers for each sub-system. We have proposed an algorithm for constructing such controllers and applied it to a numerical example.

For future investigations, a major research line is to optimize the computation of the ranking functions and to deal with robustness issues and uncertainty management. This will help substantiate the applicability of the method to more complex real-life dynamics.

Acknowledgments

This study has been carried out with financial support from the French State, managed by the French National Research Agency (ANR) in the frame of the "Investments for the future" Programme IdEx Bordeaux - SysNum (ANR-10-IDEX-03-02).

References

[1] P. Tabuada, Verification and Control of Hybrid Systems, Springer, 2009.

[2] C. Belta, B. Yordanov, E. Gol, Formal Methods for Discrete-Time Dynamical Systems, Springer, 2017.

[3] A. Girard, G. J. Pappas, Approximation metrics for discrete and continuous systems, IEEE Trans. on Aut. Cont. 52 (5) (2007) 782–798.

[4] A. Girard, G. J. Pappas, Approximate bisimulation: A bridge between computer science and control theory, European Journal of Control 17 (5) (2011) 568–578.

[5] R. Majumdar, M. Zamani, Approximately bisimilar symbolic models for digital control systems, in: P. Madhusudan, S. Seshia (Eds.), Computer Aided Verification (LNCS), Vol. 7358, Springer Berlin Heidelberg, 2012.

[6] G. Reissig, A. Weber, M. Rungger, Feedback refinement relations for the synthesis of symbolic controllers, IEEE Trans. on Aut. Cont. 62 (4) (2017) 1781–1796.

[7] M. Kloetzer, C. Belta, Dealing with nondeterminism in symbolic control, in: M. Egerstedt, B. Mishra (Eds.), Hybrid Systems: Computation and Control, Springer Berlin Heidelberg, 2008, pp. 287–300.

[8] G. Pola, P. Tabuada, Symbolic models for nonlinear control systems: Alternating approximate bisimulations, SIAM Journal on Control and Optimization 48 (2) (2009) 719–733. doi:10.1137/070698580.

[9] A. Borri, G. Pola, M. D. D. Benedetto, Symbolic models for nonlinear control systems affected by disturbances, International Journal of Control 85 (10) (2012) 1422–1432.

[10] A. Girard, G. Pola, P. Tabuada, Approximately bisimilar symbolic models for incrementally stable switched systems, IEEE Transactions on Automatic Control 55 (1) (2010) 116–126.

[11] M. Zamani, P. Esfahani, R. Majumdar, A. Abate, J. Lygeros, Symbolic control of stochastic systems via approximately bisimilar finite abstractions, IEEE Trans. on Aut. Cont. 59 (12) (2014) 3135–3150.

[12] A. Borri, G. Pola, M. D. Benedetto, A symbolic approach to the design of nonlinear networked control systems, in: Proceedings of the 15th ACM International Conference on Hybrid Systems: Computation and Control, HSCC '12, ACM, 2012, pp. 255–264.

[13] E. Dallal, P. Tabuada, On compositional symbolic controller synthesis inspired by small-gain theorems, in: 2015 54th IEEE Conference on Decision and Control, 2015, pp. 6133–6138.

[14] G. Pola, P. Pepe, M. D. Di Benedetto, Symbolic models for networks of control systems, IEEE Trans. on Aut. Cont. 61 (11) (2016) 3663–3668.

[15] G. Reissig, Computing abstractions of nonlinear systems, IEEE Trans. on Aut. Cont. 56 (11) (2011) 2583–2598.

[16] A. Girard, G. Gössler, S. Mouelhi, Safety controller synthesis for incrementally stable switched systems using multiscale symbolic models, IEEE Trans. on Aut. Cont. 61 (6) (2016) 1537–1549.

[17] A. Weber, M. Rungger, G. Reissig, Optimized state space grids for abstractions, IEEE Trans. on Aut. Cont. 62 (11) (2017) 5816–5821.
[18] M. Rungger, M. Zamani, Compositional construction of approximate abstractions of interconnected control systems, IEEE Transactions on Control of Network Systems 5 (1) (2018) 116–127.
[19] M. Mazo, A. Davitian, P. Tabuada, Pessoa: A tool for embedded controller synthesis, in: T. Touili, B. Cook, P. Jackson (Eds.), Computer Aided Verification, Springer Berlin Heidelberg, 2010, pp. 566–569.
[20] S. Mouelhi, A. Girard, G. Gössler, Cosyma: A tool for controller synthesis using multi-scale abstractions, in: Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control, HSCC '13, ACM, New York, NY, USA, 2013, pp. 83–88.
[21] T. Wongpiromsarn, U. Topcu, N. Ozay, H. Xu, R. Murray, Tulip: A software toolbox for receding horizon temporal logic planning, in: 14th Int. Conf. on Hybrid Sys.: Comput. and Control, HSCC '11, ACM, NY, USA, 2011, pp. 313–314.
[22] M. Rungger, M. Zamani, Scots: A tool for the synthesis of symbolic controllers, in: 19th Int. Conf. on Hybrid Sys.: Computation and Control, HSCC '16, ACM, NY, USA, 2016, pp. 99–104.
[23] Y. R. Stürz, A. Eichler, R. S. Smith, Distributed control design for heterogeneous interconnected systems (2020). arXiv:2004.04876.
[24] E. M. Clarke, T. Henzinger, H. Veith, R. Bloem (Eds.), Handbook of Model Checking, Springer, 2018.
[25] M. Rungger, M. Zamani, Compositional construction of approximate abstractions, in: 18th Int. Conf. on Hybrid Sys.: Computation and Control, HSCC '15, ACM, 2015, pp. 68–77.
[26] G. Pola, P. Pepe, M. D. Di Benedetto, Symbolic models for networks of discrete-time nonlinear control systems, in: American Control Conf. (ACC), 2014, pp. 1787–1792.
[27] K. Mallik, A. Schmuck, S. Soudjani, R. Majumdar, Compositional synthesis of finite state abstractions, IEEE Trans. on Aut. Cont. (2018) 1–1.
[28] W. A. Apaza-Perez, C. Combastel, A. Zolghadri, Abstraction-based low complexity controller synthesis for interconnected non-deterministic systems, in: 18th European Control Conf. (ECC), 2019, pp. 4174–4179.
[29] M. Zamani, G. Pola, M. Mazo, P. Tabuada, Symbolic models for nonlinear control systems without stability assumptions, IEEE Trans. on Aut. Cont. 57 (7) (2012) 1804–1809.
[30] A. J. van der Schaft, Equivalence of dynamical systems by bisimulation, IEEE Trans. on Aut. Cont. 49 (12) (2004) 2160–2172.
[31] N. Y. Megawati, A. van der Schaft, Bisimulation equivalence of differential-algebraic systems, International Journal of Control 91 (1) (2018) 45–56.
[32] P. J. Meyer, A. Girard, E. Witrant, Compositional abstraction and safety synthesis using overlapping symbolic models, IEEE Transactions on Automatic Control 63 (6) (2018) 1835–1841.
[33] K. Larsen, M. Mikučionis, M. Muñiz, J. Srba, J. Taankvist, Online and compositional learning of controllers with application to floor heating, in: M. Chechik, J. Raskin (Eds.), Tools and Algorithms for the Construction and Analysis of Systems, Vol. 9636, Springer, 2016, pp. 244–259.