Proceedings of the 2006 IEEE
Conference on Computer Aided Control Systems Design
Munich, Germany, October 4-6, 2006

Discrete Dynamic Feedback for a Class of Hybrid Systems on a Lattice

Domitilla Del Vecchio

Abstract— **We address the problem of designing a dynamic output feedback control for a hybrid system in order to satisfy system specifications, once the continuous variables are measured. In absence of a structure on the discrete variable space, the design of such a controller requires a number of computations at least proportional to the size of the discrete variable set and to the size of the control set. In this paper, we propose to exploit a partial order structure on the set of discrete variables and inputs. The control input is thus computed as a function of two discrete variable values that are updated at each step. This algorithm is applied to a multi-robot game involving two teams competing against each other in a "capture the flag"-like game.**

I. Introduction

Controller design problems under language specification have been extensively studied for discrete systems in the computer science literature [11]. A control perspective in the context of discrete event systems was given by [8]. The approach has been extended to specific classes of hybrid systems such as timed automata [1] and rectangular automata [13]. For these classes of hybrid systems, implementation results using tools such as [6] showed that in practice the synthesis procedure is limited to control problems with small numbers of control modes. Large part of the work on safety verification for general classes of hybrid systems has been concerned with the computation of reachable sets [12]. As noted also by [12], the problems solved with these methods are of low dimension. Also, these works are concerned with state feedback, that is, the state is available for measurement. An output map is considered in the literature of viability theory for hybrid systems [2]. However, static output feedback is usually performed.

In this paper, the continuous variables are available for measurement, thus the control and state estimation problems concern only the discrete variables. This scenario has practical interest in multi-robot systems in which the continuous variables represent the position and the velocity of a robot, while the discrete variables regulate the internal communication and coordination protocol. In [8], the control problem of discrete event systems under language specifications is considered. The proposed control algorithms with full observation have polynomial complexity in the number of states. In the case of partial observations, the control problem becomes NP complete at worse. In a practical system, the number of states can be exponential in the number of constituent processes, and therefore these control methodologies are prohibitive. Caines and Wang [5], consider the problem of steering the state of a partially observed automata to a final desired state. A dynamic programming methodology is proposed, which leads to a complexity of the control computation that is polynomial in the size of the state set, of the input set, and of the output set. Modular synthesis and special structures on the process are suggested in order to reduce computation. For example, [9] proposes algorithms for the synthesis problems of safe control policies in decentralized control of discrete event systems.

In this work, we propose a methodology based on partial order structures for computing dynamic feedback control that satisfies system specifications. This methodology relies on the state estimator on a lattice already developed in [4]. This state estimation algorithm updates two variables at each step. These are then used to compute the lower and upper bounds (in a specified lattice) of the set of inputs that satisfy the system specifications. This can be achieved under suitable order preserving assumptions of the system dynamics. A multi-robot example is proposed, which shows how to apply the proposed methodology in an attack-defense game. In particular, an attacker team runs the proposed estimation and control algorithm to design the next move based on the observation of the other team behavior. The scope of the attackers is to win the game. This scope is encoded in a system specification, which is then guaranteed by the dynamic control algorithm.

This paper is organized as follows. In section II, we introduce deterministic transition systems. In section III, we formulate the control problem of a system on a partial order. In section IV, we propose a solution to the problem. In section V, we show a multi-robot example. A small appendix revises some partial order theory and gives the proof of the main theorem.

II. Deterministic Transition Systems

A *Deterministic transition system* is a tuple $\Sigma = (\mathcal{S}, \mathcal{I}, \mathcal{Y}, F, g)$ in which $\mathcal{S}$ is a set of states, $\mathcal{Y}$ is a set of outputs, $\mathcal{I}$ is a set of inputs, $F : \mathcal{S} \times \mathcal{I} \longrightarrow \mathcal{S}$ is a transition function, and $g : \mathcal{S} \longrightarrow \mathcal{Y}$ is an output function. An *execution* of Σ is any sequence $\sigma = \{s(k)\}_{k\in\mathbb{N}}$ such that $s(0) \in \mathcal{S}$ and $s(k+1) = F(s(k), u(k))$ for $u(k) \in \mathcal{I}$ for any $k \in \mathbb{N}$. The output sequence $g(\sigma)$ is also denoted $\{y(k)\}_{k\in\mathbb{N}}$ with $y(k) = g(s(k))$. Given a system execution σ, $\sigma(k)(s)$ denotes the value of the state at step k along such an execution. Let $P : \mathcal{S} \longrightarrow \{\mathrm{T}, \mathrm{F}\}$ be a predicate on the set of states that can be either true (T) or false (F). Assume that we would like to design a control input that guarantees that such a predicate is true at any time step along a system

D. Del Vecchio is with Faculty of Electrical Engineering and Computer Science, University of Michigan ddv@umich.edu

execution starting in an initial set $X_0 \subseteq \mathcal{S}$. Since we have only output information to design such an input, we give a definition about output controllability with respect to the given predicate. Let $S = \{s \in \mathcal{S} \mid P(s) = \mathrm{T}\}$ denote the *true set* and let $Y = \{s \in \mathcal{S} \mid g(s) = y, y \in \mathcal{Y}\}$ denote the *output set*, with $Y(k) = \{s \in \mathcal{S} \mid g(s) = y(k), y(k) \in \mathcal{Y}\}$, in which $\{y(k)\}_{k\in\mathbb{N}}$ is an output sequence of Σ. For any $X \subseteq \mathcal{S}$, we use the notation $I_X(S) := \{u \in \mathcal{I} \mid F(X,u) \subset S\}$ to represent the set of inputs that map a set X inside S through F. Let $\mathcal{P}(\mathcal{S})$ denote the set of all subsets of $\mathcal{S}$, that is, $\mathcal{P}(\mathcal{S}) = \{A \mid A \subseteq \mathcal{S}\}$. The next definition proposes a concept of dynamic output feedback analogous to the one proposed by [10].

Definition 2.1: The system Σ is said to be *controllable by dynamic output feedback* with respect to true set S and initial set X_0 if there exist functions $H_1 : \mathcal{P}(\mathcal{S}) \times \mathcal{Y} \longrightarrow \mathcal{P}(\mathcal{I})$ and $H_2 : \mathcal{P}(\mathcal{S}) \times \mathcal{Y} \longrightarrow \mathcal{P}(\mathcal{S})$ such that for any execution σ and for any k, if $X(k+1) = H_2(X(k), y(k))$, $u(k) \in H_1(X(k), y(k))$, with $X(0) = X_0$, and $y(k) = g(\sigma(k)(s))$, then

(i) $s(k) \in X(k)$ with $s(k) = \sigma(k)(s)$;
(ii) $X(k) \subseteq S$.

This definition implies that there is a feedback system $\Sigma_f = (\mathcal{P}(\mathcal{S}), \mathcal{Y}, \mathcal{P}(\mathcal{I}), H_2, H_1)$ that takes as input the output of Σ, $y \in \mathcal{Y}$, has internal state $X \in \mathcal{P}(\mathcal{S})$ a subset of $\mathcal{S}$, and it has a set of inputs to Σ as its output. The set $H_1(X(k), y(k))$ is the set of inputs that map the set of states $X(k)$ inside the desired true set S. One can verify that if $I_{Y\cap S}(S)$ is nonempty for any $y \in \mathcal{Y}$ and $s(0) \in S$, system Σ is controllable by dynamic output feedback with respect to true set S and initial set $X_0 = S$. In the next section, we specialize the structure of system Σ to explicitly model the evolution of continuous and discrete variables.

III. Problem Setup

Given a deterministic transition system $\Sigma = (\mathcal{S}, \mathcal{I}, \mathcal{Y}, F, g)$, we specialize it to the case $\mathcal{S} = \mathcal{U} \times \mathcal{Z}$, in which $\mathcal{U}$ is a finite set of discrete variables denoted $\alpha \in \mathcal{U}$ and $\mathcal{Z}$ is a set of continuous variables denoted $z \in \mathcal{Z}$. The transition function will also have two components, i.e., $F = (f, h)$, in which $f : \mathcal{U} \times \mathcal{Z} \times \mathcal{I} \longrightarrow \mathcal{U}$ and $h : \mathcal{U} \times \mathcal{Z} \longrightarrow \mathcal{Z}$. The set of outputs is defined as $\mathcal{Y} = \mathcal{Z} \times \mathcal{Z}$ and the output function is $g : \mathcal{U} \times \mathcal{Z} \longrightarrow \mathcal{Y}$. For the remainder of this paper, we denote by $\Sigma = (\mathcal{U}\times\mathcal{Z}, \mathcal{I}, \mathcal{Y}, (f,h), g)$ the system represented by the following difference equations

$$\begin{aligned} \alpha(k+1) &= f(\alpha(k), z(k), u(k)) \\ z(k+1) &= h(\alpha(k), z(k)) \qquad (1) \\ (y_1(k), y_2(k)) &= (z(k), h(\alpha(k), z(k))). \qquad (2) \end{aligned}$$

Any execution of the system Σ is of the form $\sigma = \{\alpha(k), z(k)\}_{k\in\mathbb{N}}$ and the output sequence is given by $\{y(k)\}_{k\in\mathbb{N}} = \{y_1(k), y_2(k)\}_{k\in\mathbb{N}}$. Given any execution σ of the system, we will denote the values of z and α at step k in such an execution by $\sigma(k)(z)$ and $\sigma(k)(\alpha)$, respectively. At each step k, the output y has two components corresponding to the value of $z(k)$ and of $z(k+1)$.

Since the z variables are measured, we consider the problem of satisfying a predicate P on the discrete variables only. Thus, $P : \mathcal{U} \rightarrow \{\mathrm{T},\mathrm{F}\}$. Let the true set $S \subset \mathcal{U}$ be defined as $S = \{\alpha \in \mathcal{U} \mid P(\alpha) = \mathrm{T}\}$, then we consider the problem of determining an input to the system that satisfies the specification. In the present case, the output set is given by $Y = \{\alpha \in \mathcal{U} \mid y_2 = h(\alpha, y_1),\ y_1, y_2 \in \mathcal{Z}\}$.

Assume that we would like to compute the set of inputs that maintain an output set Y in S. In principle, if $\mathcal{U}$ and $\mathcal{I}$ are finite and discrete, for any $\alpha \in Y$ we can compute $f(\alpha, z, u)$ for any $u \in \mathcal{I}$ and check whether it is contained in S. Assuming that the sizes of Y and S are of the order of the size of $\mathcal{U}$, this requires a number of computations of the order of $|\mathcal{I}||\mathcal{U}|^2$.We thus propose an alternative procedure that exploits a partial order structure on the set of discrete states as well as on the set of inputs. Assume that $\alpha \in \mathbb{N}$, $Y = [2, 11]$, $S = [1, 10]$, $u \in \mathbb{Z}$, and that $f(\alpha, z, u) = f(\alpha, u) = \alpha + u$. For computing the set of inputs in $\mathbb{Z}$ such that $f([2, 11], u) \subset [1, 10]$, it is not necessary to compute f for all pairs (α, u) and check whether $f(\alpha, u) \in S$. In fact, it is enough to compute the set of $u \in \mathbb{Z}$ such that $f(2, u) \geq 1$ and the set of $u \in \mathbb{Z}$ such that $f(11, u) \leq 10$, and then intersect the two found sets, which turn out to be intervals in $\mathbb{Z}$: $[-1, \infty)$ and $(-\infty, -1]$, respectively. This simply gives the answer $u = \{-1\}$. This simplification is due to the fact that the spaces $\mathcal{U}$ and $\mathcal{I}$ are equipped with a (total in this case) order while the function f preserves such orders. This argument will be made more formal and more general in this paper by using some basic partial order theory. We next state the problem of determining a dynamic output feedback on a partial order.

Problem 1: (Dynamic Output Feedback on a Lattice)Given system $\Sigma = (\mathcal{U} \times \mathcal{Z}, \mathcal{I}, \mathcal{Y}, (f, h), g)$ with $\alpha(0) \in X_0 \subseteq S$, find a deterministic transition system $\Sigma_f = (\chi \times \chi, \mathcal{Y}, \tilde{\mathcal{I}} \times \tilde{\mathcal{I}}, (H_{21}, H_{22}), (H_{11}, H_{12}))$ with $H_{21} : \chi\times\chi\times\mathcal{Y} \rightarrow \chi$, $H_{22} : \chi\times\chi\times\mathcal{Y} \rightarrow \chi$, $H_{11} : \chi\times\chi\times\mathcal{Y} \rightarrow \tilde{\mathcal{I}}$, $H_{12} : \chi\times\chi\times\mathcal{Y} \rightarrow \tilde{\mathcal{I}}$, $(\chi, \leq)$ and $(\tilde{\mathcal{I}}, \leq)$ lattices, with $\mathcal{U} \subseteq \chi$ and $\mathcal{I} \subseteq \tilde{\mathcal{I}}$, such that if $u(k) \in [H_{11}(L(k), U(k), y(k)), H_{12}(L(k), U(k), y(k))] \cap \mathcal{I}$, and $L(k), U(k) \in \chi$ are updated by $L(k) = H_{21}(L(k-1), U(k-1), y(k-1), y(k))$ and $U(k) = H_{22}(L(k-1), U(k-1), y(k-1), y(k))$ with $\{y(k)\}_{k\geq 0} = g(\sigma)$ we have

(i) $\sigma(k)(\alpha) \in [L(k), U(k)] \cap \mathcal{U}$;
(ii) $[L(k), U(k)] \cap \mathcal{U} \subseteq S$.

The variables $L(k)$ and $U(k)$ represent the lower and upper bound of the set of possible discrete states compatible with the output sequence and with the system dynamics. The functions H_{11} and H_{12} compute the lower and upper bounds of the set of possible inputs that map the interval $[L(k), U(k)] \cap \mathcal{U}$ inside the true set S. In the next section, we propose a solution for the dynamic output feedback problem.

IV. Problem Solution

In what follows, we will say that a system Σ satisfies the dynamic controllability condition with respect to $S \subset \mathcal{U}$ if $\{u \in \mathcal{I} \mid f(S \cap Y, z, u) \subseteq S\}$ is not empty for any $y \in \mathcal{Y}$ and $z \in \mathcal{Z}$.

Definition 4.1: Let $\Sigma = (\mathcal{U} \times \mathcal{Z}, \mathcal{I}, \mathcal{Y}, (f, h), g)$, and let $(\chi, \leq)$ and $(\tilde{\mathcal{I}}, \leq)$ be lattices with $\mathcal{U} \subseteq \chi$ and $\mathcal{I} \subseteq \tilde{\mathcal{I}}$. An extension of Σ on $(\chi, \leq)$ and $(\tilde{\mathcal{I}}, \leq)$ is any system $\tilde{\Sigma} = (\chi \times$

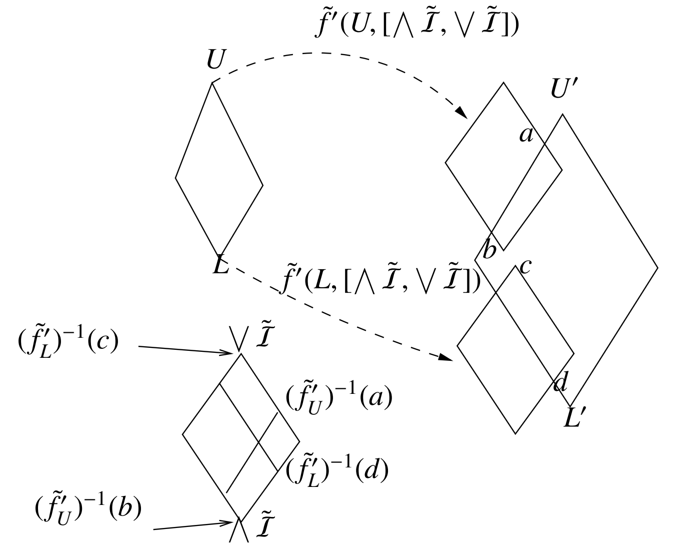
$\tilde{f}'(U,[\bigwedge\tilde{\mathcal{I}},\bigvee\tilde{\mathcal{I}}])$
U
U'
a
b
c
L
$\tilde{f}'(L,[\bigwedge\tilde{\mathcal{I}},\bigvee\tilde{\mathcal{I}}])$
$(\tilde{f}'_L)^{-1}(c)$
$\bigvee\tilde{\mathcal{I}}$
$(\tilde{f}'_U)^{-1}(a)$
d
L'
$(\tilde{f}'_L)^{-1}(d)$
$(\tilde{f}'_U)^{-1}(b)$
$\bigwedge\tilde{\mathcal{I}}$

Fig. 1. This figure explains how u_L and u_U of Lemma 4.1 are computed. The dependence on z has been omitted. Each polygon represents an interval sublattice in the partial order. u_L is the supremum ($\vee$) of $(\tilde{f}'_U)^{-1}(b)$ and $(\tilde{f}'_L)^{-1}(d)$, while u_U is the infimum ($\wedge$) of $(\tilde{f}'_L)^{-1}(c)$ and $(\tilde{f}'_U)^{-1}(a)$. Since the function $\tilde{f}'$ is an order preserving map on the interval $[L,U]$. It is enough to (1) compute the set of inputs that map L into $[L',U']$, (2) compute the set of inputs that map U into $[L',U']$, (3) intersect the these sets. Since $\tilde{f}'$ is an order isomorphism in its second argument, the sets computed in (1) and in (2) are intervals.

$\mathcal{Z},\tilde{\mathcal{I}},\mathcal{Y},(\tilde{f},\tilde{h}),\tilde{g})$ such that $\tilde{f}|_{\mathcal{U}\times\mathcal{Z}\times\mathcal{I}}=f$, $\tilde{h}|_{\mathcal{U}\times\mathcal{Z}}=h$, and $\tilde{g}|_{\mathcal{U}\times\mathcal{Z}}=g$.

Let $\tilde{f}':\chi\times\tilde{\mathcal{I}}\to\chi$ be a map with $(\chi,\le)$ and $(\tilde{\mathcal{I}},\le)$ lattices. Let $\tilde{f}'(x,[\bigwedge\tilde{\mathcal{I}},\bigvee\tilde{\mathcal{I}}])\to[\tilde{f}'(x,\bigwedge\tilde{\mathcal{I}}),\tilde{f}'(x,\bigvee\tilde{\mathcal{I}})]$ be a bijection for any $x\in\chi$. For any $w\in[\tilde{f}'(x,\bigwedge\tilde{\mathcal{I}}),\tilde{f}'(x,\bigvee\tilde{\mathcal{I}})]$ we denote by $(\tilde{f}'_x)^{-1}(w)$ the input $\tilde{u}\in[\bigwedge\tilde{\mathcal{I}},\bigvee\tilde{\mathcal{I}}]$ such that $\tilde{f}'(x,\tilde{u})=w$. Then, we have the following lemma.

Lemma 4.1: Let $(\tilde{\mathcal{I}},\le)$ and $(\chi,\le)$ be a lattices. Let $L,U\in\chi$ with $L\le U$ and $L',U'\in\chi$ with $L'\le U'$. Assume that

(i) the function $\tilde{f}':\chi\times\tilde{\mathcal{I}}\to\chi$ is order preserving on $[L,U]$ in its first argument;

(ii) $\tilde{f}':(x,[\bigwedge\tilde{\mathcal{I}},\bigvee\tilde{\mathcal{I}}])\to[\tilde{f}'(x,\bigwedge\tilde{\mathcal{I}}),\tilde{f}(x,\bigvee\tilde{\mathcal{I}})]$ is an order isomorphism for any $x\in[L,U]$;

(iii) the set $\{\tilde{u}\in\tilde{\mathcal{I}}\mid\tilde{f}'([L,U],\tilde{u})\subseteq[L',U']\}$ is not empty.

Then, $\{\tilde{u}\in\tilde{\mathcal{I}}\mid\tilde{f}'([L,U],\tilde{u})\subseteq[L',U']\}=[u_L,u_U]$, with

$$\begin{aligned}u_L &= (\tilde{f}'_L)^{-1}\left(\tilde{f}(L,\bigwedge\tilde{\mathcal{I}})\vee L'\right)\vee(\tilde{f}'_U)^{-1}\left(\tilde{f}(U,\bigwedge\tilde{\mathcal{I}})\vee L'\right)\\ u_U &= (\tilde{f}'_L)^{-1}\left(\tilde{f}(L,\bigvee\tilde{\mathcal{I}})\wedge U'\right)\wedge(\tilde{f}'_U)^{-1}\left(\tilde{f}(U,\bigvee\tilde{\mathcal{I}})\wedge U'\right).\end{aligned}$$

The proof of this lemma is pictorially shown in Figure 1.

Let $\tilde{\Sigma}=(\chi\times\mathcal{Z},\mathcal{I},\mathcal{Y},(\tilde{f},\tilde{h}),\tilde{g})$ be an extension of Σ on $(\chi,\le)$. We will assume that the output set of the extended system is an interval, that is, $\tilde{Y}=\{x\in\chi\mid y_2=\tilde{h}(x,y_1),\ (y_1,y_2)\in\mathcal{Y}\}=[\bigwedge\tilde{Y},\bigvee\tilde{Y}]$. For an output sequence $\{y(k)\}_{k\ge0}$, we will denote the output set by $\tilde{Y}(k)$ and its lower and upper bounds by $\bigwedge Y(k)=L_y(k)$ and $\bigvee Y(k)=U_y(k)$. We also assume that $\tilde{S}\subset\chi$ is a set such that $\tilde{S}\cap\mathcal{U}=S$ and that $\tilde{\Sigma}$ satisfies the dynamic output controllability condition with respect to $\tilde{S}$. This implies by the definition of $\tilde{f}$ that Σ satisfies the dynamic output controllability condition with respect to S.

Definition 4.2: Let $\tilde{\Sigma}=(\chi\times\mathcal{Z},\mathcal{I},\mathcal{Y},(\tilde{f},\tilde{h}),\tilde{g})$ be an extension of Σ on $(\chi,\le)$. Let $(\tilde{\mathcal{I}},\le)$ be a lattice with $\mathcal{I}\subseteq\tilde{\mathcal{I}}$. Let $[L,U]$ be any interval in $\tilde{S}\cap\tilde{Y}$. If there are a function $\tilde{f}':\chi\times\tilde{\mathcal{I}}\to\chi$ such that $\tilde{f}':(x,[\bigwedge\tilde{\mathcal{I}},\bigwedge\tilde{\mathcal{I}}])\to[\tilde{f}'(x,\bigwedge\tilde{\mathcal{I}}),\tilde{f}'(x,\bigvee\tilde{\mathcal{I}})]$ is an order isomorphism for any $x\in\chi$ and an order preserving map in the first argument, constants $L^*,U^*\in[L,U]$, and $L',U'\in\tilde{S}$ such that $\{u\in\mathcal{I}\mid\tilde{f}([L,U],z,u)\subseteq\tilde{S}\}\supseteq\mathcal{I}\cap\{\tilde{u}\in\tilde{\mathcal{I}}\mid\tilde{f}'([L^*,U^*],\tilde{u})\subseteq[L',U']\}$, with the righthand set not empty, then the extension $\tilde{\Sigma}$ of Σ is said to admit an order compatible input extension on $(\tilde{\mathcal{I}},\le)$ with respect to $\tilde{S}$. The tuple $(\tilde{f}',L^*,U^*,L',U')$ is named an order compatible tuple for $[L,U]$.

The reason for introducing this definition is that while the computation of the set $\{u\in\mathcal{I}\mid\tilde{f}([L,U],z,u)\subseteq\tilde{S}\}$ might be hard, the computation of the set $\{\tilde{u}\in\tilde{\mathcal{I}}\mid\tilde{f}'([L^*,U^*],\tilde{u})\subseteq[L',U']\}$ is simpler by virtue of Lemma 4.1.

Definition 4.3: The system extension $\tilde{\Sigma}$ is said to be *output interval compatible* with respect to $\tilde{S}$ if for any $[L,U]\subset\chi$, we have that $\tilde{f}([L,U],z,u)\subseteq\tilde{S}$ implies $[\bigwedge(\tilde{f}([L,U],z,u)\cap\tilde{Y}),\bigvee(\tilde{f}([L,U],z,u)\cap\tilde{Y})]\subseteq\tilde{S}$ for any $y\in\mathcal{Y}$ and $z\in\mathcal{Z}$.

In this paper, we are not making any order preserving assumption on the function f. It is in fact always possible to break a function into order preserving pieces on an interval. Let $f:\chi\to\chi$ with $(\chi,\le)$ a lattice. Let $[L,U]\subseteq\chi$ be an interval. The function $\tilde{f}$ is said to be a piecewise order isomorphism on $[L,U]$ if there are a finite number of points $L^j,U^j\in[L,U]$ such that $[L,U]=[L^1,U^1]\cup\ldots\cup[L^M,U^M]$ with $[L^i,U^i]\cap[L^j,U^j]=\emptyset$ for $i\neq j$ and such that $f:[L^j,U^j]\to[f(L^j),f(U^j)]$ is an order isomorphism for any j. The points L^j are said lower knot points and the point U^j are said upper knot points of f in $[L,U]$. On a finite discrete set, a map can always be broken into order isomorphisms: in the limit, each interval will contain only one element.

The next theorem provides a solution to Problem 1.

Theorem 4.1: Let $\tilde{\Sigma}$ be an extension of Σ on $(\chi,\le)$. Let $\alpha(0)$ be such that $\tilde{Y}(0)\subseteq\tilde{S}$. Assume that $\tilde{\Sigma}$ admits an order compatible input extension on $(\tilde{\mathcal{I}},\le)$ with respect to $\tilde{S}$. Assume that $\tilde{\Sigma}$ is output interval compatible with respect to $\tilde{S}$. Then,

$$\begin{aligned}L(k+1) &= \bigwedge_{\bar{L}^j\le\bar{U}^j}\bar{L}^j,\ \bar{L}^j=\tilde{f}(L^j(k),z(k),u(k))\vee L_y(k+1)\\ U(k+1) &= \bigvee_{\bar{L}^j\le\bar{U}^j}\bar{U}^j,\ \bar{U}^j=\tilde{f}(U^j(k),z(k),u(k))\wedge U_y(k+1),\end{aligned}$$

with $L^j(k),U^j(k)$ knot points of $\tilde{f}$ on $[L(k),U(k)]$, $L(0)=L_y(0)$, $U(0)=U_y(0)$, and $u(k)\in[u_L(k),u_U(k)]\cap\mathcal{I}$ with $u_L(k)=(\tilde{f}')^{-1}_{L^*(k)}(\tilde{f}'(L^*(k),\bigwedge\tilde{\mathcal{I}})\vee L'(k))\vee(\tilde{f}')^{-1}_{U^*(k)}(\tilde{f}'(U^*(k),\bigwedge\tilde{\mathcal{I}})\vee L'(k))$ and $u_U(k)=(\tilde{f}')^{-1}_{L^*(k)}(\tilde{f}'(L^*(k),\bigvee\tilde{\mathcal{I}})\wedge U'(k))\wedge(\tilde{f}')^{-1}_{U^*(k)}(\tilde{f}'(U^*(k),\bigvee\tilde{\mathcal{I}})\wedge U'(k))$, with $(\tilde{f}',L^*(k),U^*(k),L'(k),U'(k))$ an order compatible tuple for $[L(k),U(k)]$, solve Problem 1.

From the update laws of $L(k)$ and $U(k)$, it is clear that the amounts of computation increases with the number of order isomorphisms in which $\tilde{f}$ needs to be broken. In the next section, we propose a multi-robot game to show how

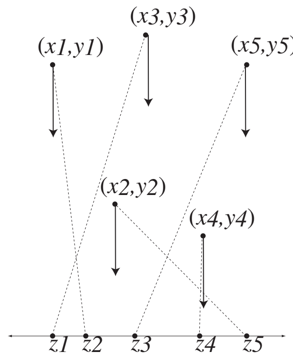

Fig. 2. Example with five robots per team.

to apply Theorem 4.1.

V. Example: A Multi-robot System

We consider a simplified version of the game called "RoboFlag Drill" already considered in [4].

A. System description

Some number of robots with positions $(z_i, 0) \in \mathbb{R}^2$, which we refer to as blue robots, must defend their zone $\{(x, y) \in \mathbb{R}^2 \mid y \leq 0\}$ from an equal number of incoming robots, which we refer to as red robots with positions $(x_i, y_i) \in \mathbb{R}^2$. An example with five robots is illustrated in Figure 2.

Let N represent the number of robots in each team. The robots start with an arbitrary (bijective) assignment $\alpha : \{1, ..., N\} \rightarrow \{1, ..., N\}$, where α_i is the red robot that blue robot i is required to intercept. At each step, each blue robot communicates with its neighbors and decides to either switch assignments with its left or right neighbor or keep its assignment. In the case in which the red robots are just coming down straight, it is possible to show that the α assignment reaches the equilibrium value $(1, ..., N)$ [7]. In this work, we consider the problem of estimating the current assignment α given the motions of the blue robots, which is then used by the red robots to determine a strategy of attack. We thus assume that the red robots can also move horizontally by exchanging position with a close red robot. The RoboFlag Drill system can be specified as follows

$$y_i(k+1) = y_i(k) - \delta \quad \text{if} \quad y_i(k) \geq \delta \tag{3}$$

$$z_i(k+1) = z_i(k) + \delta \quad \text{if} \quad z_i(k) < x_{\alpha_i(k)} \tag{4}$$

$$z_i(k+1) = z_i(k) - \delta \quad \text{if} \quad z_i(k) > x_{\alpha_i(k)} \tag{5}$$

$$\begin{aligned}(\alpha_i(k+1), \alpha_{i+1}(k+1)) = (\alpha_{i+1}(k), \alpha_i(k)) \text{ if}\\ x_{\alpha_i(k)} \geq z_{i+1}(k) \wedge x_{\alpha_{i+1}(k)} \leq z_{i+1}(k),\end{aligned} \tag{6}$$

$$\begin{aligned}(x_i(k+1), x_j(k+1)) = (x_j(k), x_i(k)) \text{ if}\\ \lambda_j(k) = \lambda_i(k) + 1 \text{ and } swap_{i,j}(k) = 1,\end{aligned} \tag{7}$$

$$\begin{aligned}(\lambda_i(k+1), \lambda_j(k+1)) = (\lambda_j(k), \lambda_i(k)) \text{ if}\\ \lambda_j(k) = \lambda_i(k) + 1 \text{ and } swap_{i,j}(k) = 1.\end{aligned} \tag{8}$$

Equation (6) establishes that two blue robots trade their assignments if the current assignments cause them to go toward each other. Condition (7) allows two adjacent red robots to swap their location. Here, λ_i is the location of red robot i. Locations are $\{1, ..., N\}$ and denote the order along the x direction in which the red robots are displaced. We start the game with $\lambda = (1, ..., N)$, $z_i \leq z_{i+1}$, and $x_i < z_i < x_{i+1}$. The variable $swap_{i,j}$ is the control input to the system represented by equations (4–6).

For the purpose of control, we define the system $\Sigma = (\mathcal{U} \times \mathcal{Z}, \mathcal{I}, \mathcal{Y}, (f, h), g)$ that models the dynamics of the blue robots with inputs $swap_{i,j}$ (4–6) as follows. Let $\bar{\alpha}_i = \lambda_{\alpha_i}$ represent the assignment of blue robot i to one of the locations. Let $\bar{x}_i = x_i(0)$ represent the x coordinate of location i. We let the variable $u_i = 1$ if and only if the red robot at location i swaps with the red robot at location $i + 1$. Then, set $\mathcal{U} = \text{perm}(N)$, with $\bar{\alpha} \in \mathcal{U}$, $\mathcal{Z} = \mathbb{R}^N$, $\mathcal{I} = \{u \in \{-1, 0, 1\}^N \mid u_i = 1 \Leftrightarrow u_{i+1} = -1$, and $u_N \neq 1$ and $u_1 \neq -1\}$, $\mathcal{Y} = \mathbb{R}^N \times \mathbb{R}^N$. The functions are defined as follows: $f(\bar{\alpha}, z, u) = G(F(\bar{\alpha}, z), u)$, in which $F(\bar{\alpha}, z)$ is represented by relations (6) with $x_i(k) = x_i(0) = \bar{x}_i$ and α_i replaced by $\bar{\alpha}_i$ and $G(\beta, u) = \beta'$, with $u_j = 1 \Rightarrow$ (if $\beta_i = j \Rightarrow \beta'_i = j + 1$) and (if $\beta_i = j + 1 \Rightarrow \beta'_i = j$). Note that we also will use the following notation $G(\beta, u) = (G_1(\beta_1, u), ..., G_N(\beta_N, u))$, in which $G_i(\beta_i, u) = \beta'_i$, with $u_j = 1 \Rightarrow$ (if $\beta_i = j \Rightarrow \beta'_i = j + 1$) and (if $\beta_i = j + 1 \Rightarrow \beta'_i = j$). We assume for simplicity that $z_i \leq z_{i+1}$ and $\bar{x}_i < z_i < \bar{x}_{i+1}$ for all k. In the sequel, with abuse of notation we will remove the bar and denote $\bar{\alpha}$ and $\bar{x}$ by α and x, respectively. The function $h(\alpha, z)$ is represented by relations (4-5).

Let the entropy of the blue robots be defined by $E(\alpha) = \frac{1}{2}\sum_{i=1}^{N} |\alpha_i - i|$. In absence of any input to the system (i.e. $u(k) = 0$ for any k), this entropy converges to zero. We define the control specification $P : \mathcal{U} \rightarrow$ {T,F} to be $P(\alpha) = \text{T}$ if $E(\alpha) \geq 2$ and $P(\alpha) = \text{F}$ otherwise. In words, we want to solve the following problem: Given measurements z determine a dynamic control input $u(k)$ such that $E(\alpha(k)) \geq 2$ for all $k \in \mathbb{N}$. Having entropy greater or equal than two implies that two or more swappings of assignments among the blue robots are needed before the assignment reaches the value $(1, ..., N)$. One can verify that $S = \{\alpha \mid E(\alpha) \geq 2\} = \{\alpha \mid \exists\, i, j$, with $j > i + 1$ such that $\alpha_i \neq i$ and $\alpha_j \neq j\}$.

In the next section, we show how to use the state estimation and control computation on a partial order for the solution of the described control problem.

B. Dynamic control on a partial order

In this section, we introduce the partial order $(\chi, \leq)$ and the set $\tilde{S}$ extension of S in χ. Let $(\chi, \leq)$ be the partial order with $\chi = \mathbb{N}^N$ with order established componentwise. For any set $X \subseteq \mathbb{N}^N$, we denote $[X]_j$ the projection along j of such set. Then, we have the following characterization of the set $\tilde{S}$. A set $\tilde{S} \subseteq \chi$ with $S = \tilde{S} \cap \mathcal{U}$ is given by $\tilde{S} = \cup_i \tilde{S}_i$, in which $\tilde{S}_i$ for each i are intervals of four types: (a) there are $l < j$ such that $[\tilde{S}_i]_l = [l + 1, N]$ and $[\tilde{S}_i]_j = [j + 1, N]$; (b) there are $l < j$ such that $[\tilde{S}_i]_l = [1, l - 1]$ and $[\tilde{S}_i]_j = [j + 1, N]$; (c) there are $l < j$ with $j > l + 1$ such that $[\tilde{S}_i]_l = [l + 1, N]$ and $[\tilde{S}_i]_j = [1, j - 1]$; (d) there are $l < j$ such that $[\tilde{S}_i]_l = [1, l - 1]$ and $[\tilde{S}_i]_j = [1, j - 1]$. One can easily verify by inspection that such a constructed $\tilde{S}$ is such that $\tilde{S} \cap \mathcal{U} = S$.

Let us define the extension $\tilde{F} : \chi \times \mathcal{Z} \to \chi$ as F with now $\alpha \in \mathbb{N}^N$. Clearly, $\tilde{F}|_{\mathcal{U}\times\mathcal{Z}} = F$. Also, we define $\tilde{h} : \chi \times \mathcal{Z} \to \mathcal{Z}$ as h with $\alpha \in \mathbb{N}^N$. Also, we have that $\tilde{h}|_{\mathcal{U}\times\mathcal{Z}} = h$. Let us denote $\tilde{Y} = \{x \in \chi \mid z' = \tilde{h}(x, z)\}$. One can easily check that this set is an interval for any pair z, z'. Also, the function $\tilde{F}$ is an order isomorphism on the set $\tilde{Y}$ (more details can be found in [4]). The function $\tilde{G} : \chi \times \mathcal{I} \to \chi$ is defined as G in which the first argument now belongs to $\mathbb{N}^N$. Then $\tilde{f} = \tilde{G} \circ \tilde{F}$, in which one can check that $\tilde{f}|_{\mathcal{U}\times\mathcal{Z}\times\mathcal{I}} = f$. We thus have the extended system $\tilde{\Sigma} = (\chi \times \mathcal{Z}, \mathcal{Y}, \mathcal{I}, (\tilde{f}, \tilde{h}), \tilde{g})$. One can verify that $\tilde{Y} \cap \tilde{S}$ is an interval in χ for any y and that if P is an interval, also $\tilde{F}(P, z)$ is an interval. It is easy to show that the extended system $\tilde{\Sigma} = (\chi \times \mathcal{Z}, \mathcal{I}, \mathcal{Y}, (\tilde{f}, \tilde{h}), \tilde{g})$ satisfies the dynamic controllability condition with respect to $\tilde{S}$ if $N > 4$, that is, for $P = \tilde{Y} \cap \tilde{S}$ there is $u \in \mathcal{I}$ such that $\tilde{f}(P, z, u) \subseteq \tilde{S}$.

The set $\tilde{S}$ is given by the union of a large number of intervals. At each step, we determine a particular subset $\tilde{X} \subset \tilde{S}$ in which we want (and for which it is possible) to keep the system state. The following algorithm, computes $\tilde{X}$ componentwise.

Algorithm 5.1: Let $P \subseteq \tilde{Y} \cap \tilde{S}$ be an interval, and let $P' = \tilde{F}(P, z)$.

```
Initialize $\tilde{X}_i = [1, N]$, $flag_i = 0$ for all $i$
For $i = 1 : N$
    If $\min(P'_i) = i$ and $flag_{i-1} = 0 \Longrightarrow \tilde{X}_i = [1, i-1] \cup [i+1, N]$ and $flag_i = 1$
End
For $i = 2 : N$
    If $\max(P'_{i-1}) = i-1$ and $flag_{i-1} = 0 \Longrightarrow \tilde{X}_{i-1} = [1, i-2] \cup [i, N]$ and $flag_i = 1$
End
For $i = 1 : N$
    If $\min(P'_i) \geq i+1$ and $flag_{i+1} = 0 \Longrightarrow \tilde{X}_i = [i+1, N]$
    If $\max(P'_i) \leq i-1$ and $flag_i = 0 \Longrightarrow \tilde{X}_i = [1, i-1]$
End.
```

One can show that the set $\tilde{X} = \tilde{X}_1 \times ... \times \tilde{X}_N$ is contained in the set $\tilde{S}$ and that there exist an input $u \in \mathcal{I}$ that maps the interval $P \subseteq \tilde{S} \cap \tilde{Y}$ into $\tilde{X}$. Since $\tilde{X}$ depends on the set P of Algorithm 5.1, we will use the notation $\tilde{X}(P)$. Thus, $\{u \in \mathcal{I} \mid \tilde{f}(P, z, u) \subseteq \tilde{X}(P)\}$ is nonempty for any interval $P \subseteq \tilde{Y} \cap \tilde{S}$ and it is contained in the set $\{u \in \mathcal{I} \mid \tilde{f}(P, z, u) \subseteq \tilde{S}\}$. The next proposition shows that system $\tilde{\Sigma}$ admits an order compatible input extension on $(\tilde{\mathcal{I}}, \leq)$, where $\tilde{\mathcal{I}} = \{-1, 0, 1\}^N$.

Proposition 5.1: Let $P \subseteq \tilde{Y} \cap \tilde{S}$ be an interval, let $P' = \tilde{F}(P, z)$, and let $\tilde{X}(P)$ as computed by Algorithm 5.1. Then, $\{u \in \mathcal{I} \mid \tilde{G}(P', u) \subseteq \tilde{X}(P)\} = \mathcal{I} \cap \{\tilde{u} \in \tilde{\mathcal{I}} \mid \tilde{f}'([L^*, U^*], u) \subseteq [L', U']\}$ with $\tilde{\mathcal{I}} = \{-1, 0, 1\}^N$, in which

(i) $L^*_i = U^*_i = i$ for all i such that $\tilde{X}_i = [1, i-1] \cup [i+1, N]$ or $\tilde{X}_i = [1, N]$, $L^*_i = U^*_i = i-1$ if $\tilde{X}_i = [1, i-1]$, and $L^*_i = U^*_i = i+1$ if $\tilde{X}_i = [i+1, N]$;

(ii) $\tilde{f}'([L^*, U^*], u) = (\tilde{f}'_1([L^*_1, U^*_1], u_{k_1}), ..., \tilde{f}'_N([L^*_N, U^*_N], u_{k_N})$ with $u_{k_i} = u_i$ if $\tilde{X}_i = [1, i-1] \cup [i+1, N]$ or if $\tilde{X}_i = [1, N]$, $u_{k_i} = u_{i-1}$ if $\tilde{X}_i = [1, i-1]$, and $u_{k_i} = u_{i+1}$ if $\tilde{X}_i = [i+1, N]$;

(iii) $\tilde{f}'_i(x_1, x_2) = x_1 + x_2$ for $x_1 \in [1, N]$ and $x_2 \in \{-1, 0, 1\}$;

(iv) $[L'_i, U'_i] = [1, i-1]$ if $P'_i = [i, N]$ and $\tilde{X}_i = [1, i-1] \cup [i+1, N]$, or if $\tilde{X}_i = [1, i-1]$; $[L'_i, U'_i] = [i+1, N]$ if $P'_i = [1, i]$ and $\tilde{X}_i = [1, i-1] \cup [i+1, N]$, or if $\tilde{X}_i = [i+1, N]$; $[L'_i, U'_i] = [1, N]$ if $\tilde{X}_i = [1, N]$.

Clearly, $\tilde{f}'$ is an order isomorphism in the second argument for any $x \in [L^*, U^*]$ and order preserving in the first argument. This along with $\{u \in \mathcal{I} \mid \tilde{f}(P, z, u) \subseteq \tilde{X}(P)\} \subseteq \{u \in \mathcal{I} \mid \tilde{f}(P, z, u) \subseteq \tilde{S}\}$ with $P \subseteq \tilde{Y} \cap \tilde{S}$ implies that system $\tilde{\Sigma}$ admits an order compatible input extension. The next proposition provides the intervals on which $\tilde{G}$ is an order isomorphism.

Proposition 5.2: For any $u \in \mathcal{I}$ and any $[L, U] \in \chi$, let $[L, U] = [L_1, U_1] \times ... \times [L_N, U_N]$ with $[L_i, U_i] = [L^1_i, U^1_i] \cup ... \cup [L^{M_i}_i, U^{M_i}_i]$ and $[L^j_i, U^j_i] \cap [L^k_i, U^k_i] = \emptyset$. Assume that the sets $[L^j_i, U^j_i]$ for any j satisfy the following conditions

(i) if $u_j = 1$ then $j \in [L_i, U_i] \Longrightarrow L^p_i = U^p_i = j$ for some p, $j+1 \in [L_i, U_i] \Longrightarrow L^p_i = U^p_i = j+1$ for some p;

(ii) if $u_j = -1$ then $j \in [L_i, U_i] \Longrightarrow L^p_i = U^p_i = j$ for some p, $j-1 \in [L_i, U_i] \Longrightarrow L^p_i = U^p_i = j-1$ for some p.

Then, $\tilde{G}([L^J, U^J], u) \Longrightarrow [\tilde{G}(L^J, u), \tilde{G}(U^J, u)]$ is an order isomorphism for any $L^J \leq U^J$ with $L^J = (L^{j_1}_1, ..., L^{j_M}_N)$ and $U^J = (U^{j_1}_1, ..., U^{j_M}_N)$ for $j_i \in \{1, ..., M_i\}$.

The proof of this proposition is apparent once one notices that $u_j = 1$ leads to a swapping of j with $j+1$ in each coordinate set $[L_i, U_i]$. The next proposition shows that system $\tilde{\Sigma}$ is output interval compatible with respect to $\tilde{S}$.

Proposition 5.3: Let $\tilde{Y} = [L_y, U_y]$ and $[L, U] \subseteq \chi$. Assume $\tilde{G}([L, U], u) \subseteq \tilde{X} \subset \tilde{S}$ and let $[L, U] = [L_1, U_1] \times ... \times [L_N, U_N]$ with $[L_i, U_i] = [L^1_i, U^1_i] \cup ... \cup [L^{M_i}_i, U^{M_i}_i]$ and the sets $[L^j_i, U^j_i]$ as in Proposition 5.2. Then, we have that $\tilde{G}_i([L_i, U_i], u) \cap [L_{y,i}, U_{y,i}] \subseteq \tilde{X}_i$ for all i.

The next section shows simulation results obtained by constructing the lower-upper bound estimator of Theorem 4.1, which is used to compute an allowed input to the system.

C. Simulation results

We consider the system starting with a random (unknown) assignment $\alpha(0) \in S$ such that $\tilde{Y}(0) \subset \tilde{S}$, which implies that the initial entropy of the assignment is larger or equal than

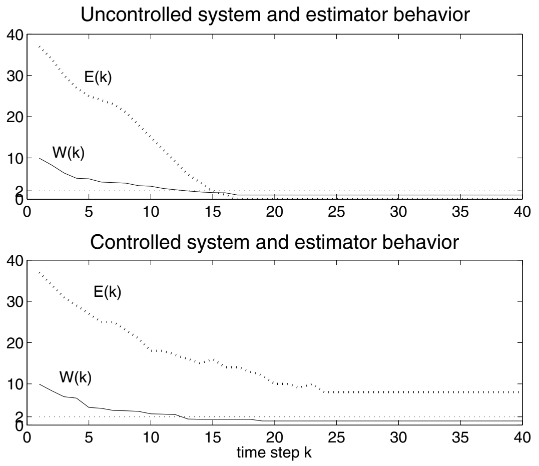

Fig. 3. Convergence plots of the estimator and of the entropy for $N = 15$ and $\alpha(0) = (4, 8, 9, 2, 13, 15, 6, 5, 12, 10, 1, 14, 3, 7, 11)$. The horizontal dotted lines corresponds to $E(k) = 2$.

two. Given $P \subseteq \chi$, which at the first step is equal to $\tilde{Y}(0)$ and in general it is equal to $[L(k), U(k)]$, we compute by means of Algorithm 5.1 the set $\tilde{X}(P)$ in which we want to map the state at the next step. Once $\tilde{X}(P)$ is computed, we use Proposition 5.1 to compute an order compatible tuple. Then, we use Lemma 4.1 to compute $u_L(k)$ and $u_U(k)$, that is, the lower and the upper bounds of the set of possible inputs. Among these inputs, we choose the one that has the largest number of zero components (smallest control action). We thus apply this input to the system. Then, we update $[L(k), U(k)]$ by exploiting Propositions 5.2 and 5.3 for using the update laws of Theorem 4.1.

In Figure 3, we report convergence plots of the estimator and of the entropy for $N = 15$ robots. We have that $W(k) = 1/N \sum_{i=1}^{N} |m_i(k)|$, in which $m_i(k)$ is the coordinate set $[L_i(k), U_i(k)]$ minus all the singletons that occur at other coordinates. The controlled system has always $E(k) \geq 2$. The estimator with no input applied to the system not always converges before the state $\alpha(k)$ has reached $E(k) = 2$. Thus, a strategy that runs the estimator first, and only when it has converged the controller is applied to the system does not guarantee the specification. The computation requirement for the implementation of the dynamic controller is proportional to N, that is, to the number of variables that need to be controlled. If we had not used any structure, we would have incurred in a number of computations at each step at least of the order of $(N!)^2$.

VI. Conclusions and Future Work

In this work, we have shown how to exploit a partial order structure on the set of inputs and of discrete states in order to design a low-computation dynamic controller that keeps the system state in a desired set. We showed how to apply the proposed algorithm to a multi-robot system involving two teams competing against each other in order to design a dynamic winning controller for one of the teams.

We conjecture that, as in the case of state estimation (see [4]), also for the dynamic control problem if the system is controllable by dynamic output feedback it is always possible to find suitable partial orders and system extensions for which the assumptions are verified. This point however needs more investigation. The proposed method does not rely on enumeration and exhaustive search, and therefore it is in principle applicable to state estimation and control of continuous states. This will be studied in future work. Finally, we would like to apply this approach to more complex multi-agent systems scenarios in which the agents can move in a plane as opposed to a line with more complex dynamics.

References

[1] E. Asarin, O. Maler, and A. Pnueli. Symbolic controller design for discrete and timed systems. *Hybrid Systems: Computation and Control,* Lecture Notes in Computer Science, volume 999, P. Antsakilis, W. Kohn, A. Nerode, and S. Sastry, Eds. Springer Verlag, pages 1–20, 1995.

[2] J. Aubin, J. Lygeros, M. Quincampoix, S. Sastry, and N. Seube. Impulse differential inclusions: A viability approach to hybrid systems. *IEEE Transactions on Automatic Control*, 47(1):2–20, 2002.

[3] B. A. Davey and H. A. Priesteley. *Introduction to Lattices and Order*. Cambridge University Press, 2002.

[4] D. DelVecchio, R. M. Murray, and E. Klavins. Discrete state estimators for systems on a lattice. *Automatica*, 42(2):271–285, 2006.

[5] P. E.Caines and S. Wang. Classical and logic based regulator design and its complexity for partially observed automata. In *Conf. on Decision and Control*, pages 132–137, 1989.

[6] T. A. Henzinger, P. H. Ho, and H. Wong-Toi. A user guide to HYTECH. In *TACAS 95: Tools and Algorithms for the construction and analysis of systems,* Lecture Notes in Computer Science, vol. 1019, E. Brinksma, W. Cleaveland, K. Larsen, T. Margaria, and B. Steffen, Eds. Springer-Verlag, pages 41–71, 1995.

[7] E. Klavins and R. M. Murray. Distributed algorithms for cooperative control. *Pervasive Computing*, 3:56–65, 2004.

[8] P. J. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–98, 1989.

[9] K. Rohloff and S. Lafortune. On the synthesis of safe control policies in decentralized control of discrete event systems. *IEEE Transactions on Automatic Control*, 48(6):1064–1068, 2003.

[10] E. D. Sontag. *Mathematical Control Theory*. Springer, 1998.

[11] W. Thomas. On the synthesis of strategies in infinite games. In *Proceedings of the STACS 95,* E. W. Mayr and C. Puech, Eds. Springer Verlag, pages 1–13, 1995.

[12] C. J. Tomlin, I. Mitchell, A. M. Bayen, and M. Oishi. Computational techniques for the verification of hybrid systems. *Proceedings of the IEEE*, 91(7):986–1001, 2003.

[13] H. Wong-Toi. The synthesis of controllers for linear hybrid automata. In *Conf. on Decision and Control*, pages 4607–4613, 1997.

A. Appendix

A partial order is a set χ with a partial order relation "$\leq$", and we denote it by the pair $(\chi, \leq)$. For any $x, w \in \chi$, the $\sup\{x, w\}$ is the smallest element that is larger than both x and w. In a similar way, the $\inf\{x, w\}$ is the largest element that is smaller than both x and w. We define the *join* "$\vee$" and the *meet* "$\wedge$" of two elements x and w in χ as (1) $x \vee w = \sup\{x, w\}$ and $x \wedge w = \inf\{x, w\}$; (2) if $S \subseteq \chi$, $\bigvee S = \sup\ S$ and $\bigwedge S = \inf\ S$. Let $(\chi, \leq)$ be a lattice and let $S \subseteq \chi$ be a non-empty subset of χ. Then, $(S, \leq)$ is a *sublattice* of χ if $a, b \in S$ implies that $a \vee b \in S$ and $a \wedge b \in S$. Any interval sublattice of $(\chi, \leq)$ is given by $[L, U] = \{w \in \chi \mid L \leq w \leq U\}$ for $L, U \in \chi$. That is, this special sublattice can be represented by only two elements. Let $(P, \leq)$ and $(Q, \leq)$ be partially ordered sets. A map $f : P \to Q$ is (i) an *order preserving map* if $x \leq w \implies f(x) \leq f(w)$; (ii) an *order embedding* if $x \leq w \iff f(x) \leq f(w)$; (iii) an *order isomorphism* if it is order embedding and it maps P *onto* Q. For more details, the reader is referred to [3].

Proof: [Proof of Theorem 4.1]. For any k, if $[L(k), U(k)] \subseteq \tilde{S} \cap \tilde{Y}(k)$, we have $\{u \in \mathcal{I} \mid \tilde{f}([L(k), U(k)], u) \subseteq \tilde{S}\} \supseteq \mathcal{I} \cap \{\tilde{u} \in \tilde{\mathcal{I}} \mid \tilde{f}'([L^*(k), U^*(k)], \tilde{u}) \subseteq [L'(k), U'(k)]\}$, in which the righthand side is not empty for $(\tilde{f}', L^*(k), U^*(k), L'(k), U'(k))$ an order compatible tuple for $[L(k), U(k)]$. As a consequence, we can apply Lemma 4.1 to obtain $u_L(k)$ and $u_U(k)$. We need to show that $[L(k), U(k)] \subseteq \tilde{S} \cap \tilde{Y}(k)$ for any k. We proceed by induction argument on k. We omit the dependence of $\tilde{f}$ on z for simplicity. By assumption, we have that $[L(0), U(0)] = \tilde{Y}(0) \subset \tilde{S}$. Let us then assume that $[L(k), U(k)] \subseteq \tilde{Y}(k) \cap \tilde{S}$ and show that also $[L(k+1), U(k+1)] \subseteq \tilde{Y}(k+1) \cap \tilde{S}$. Since $[L(k), U(k)] \subseteq \tilde{Y}(k) \cap \tilde{S}$, then $\tilde{f}([L(k), U(k)], u) \subseteq \tilde{S}$ for $u \in [u_L(k), u_U(k)] \cap \mathcal{I}$. Let $L^i(k), U^i(k)$ be knot points of $\tilde{f}$ on $[L(k), U(k)]$. Then, we have that $\tilde{f}([L(k), U(k)], u) \cap \tilde{Y}(k+1) = [\tilde{f}(L^1(k), u) \vee L_y(k+1), \tilde{f}(U^1(k), u) \wedge U_y(k+1)] \cup \ldots \cup [\tilde{f}(L^M(k), u) \vee L_y(k+1), \tilde{f}(U^M(k), u) \wedge U_y(k+1)]$. It follows that $\bigwedge(\tilde{f}([L(k), U(k)], u) \cap \tilde{Y}(k+1)) = \bigwedge_{i \in I} \tilde{f}(L^i(k), u) \vee L_y(k+1) = L(k+1)$ and $\bigvee(\tilde{f}([L(k), U(k)], u) \cap \tilde{Y}(k+1)) = \bigvee_{i \in I} \tilde{f}(U^i(k), u) \wedge U_y(k+1) = U(k+1)$ with $I = \{i \mid \tilde{f}(L^i(k), u) \vee L_y(k+1) \leq \tilde{f}(U^i(k), u) \wedge U_y(k+1)$ and $u \in [u_L(k), u_U(k)] \cap \mathcal{I}\}$. By the output interval compatibility assumption, $[\bigwedge(\tilde{f}([L(k), U(k)], u) \cap \tilde{Y}(k+1)), \bigvee(\tilde{f}([L(k), U(k)], u) \cap \tilde{Y}(k+1)] \subseteq \tilde{S}$, thus we also have that $[L(k+1), U(K+1)] \subseteq \tilde{S}$. Also, $\alpha(k) \in [L(k), U(k)] \cap \mathcal{U}$ by construction, which shows (i) of Problem 1. Since $[L(k), U(k)] \subseteq \tilde{S}$ and $\tilde{S} \cap \mathcal{U} = S$, we also have that $[L(k), U(k)] \cap \mathcal{U} \subseteq S$, which shows (ii) of Problem 1. ■