Safety Control Synthesis for Systems with Missing Measurements

Liren Yang and Necmiye Ozay

Electrical Engineering and Computer Science Department,
University of Michigan, Ann Arbor, MI 48109 USA
(e-mail: yliren,necmiye@umich.edu).

Abstract: We study the problem of synthesizing controllers that enforce hard state-constraints under occasionally missing state measurements. Although the problem can be solved as a partial information game via power set construction, it is undesired to do so because power set construction is computationally expensive. In this paper, we consider the instances where the missing data pattern can be specified by an automaton, and show that, for such cases, the synthesis problem admits a simpler solution. We introduce string automata and exploit the causality requirement to define the structure of the controller, and further use this structure to derive a product system. The synthesis problem then reduces to a full-information game on this product system. Two examples illustrate the efficacy of the proposed approach.

Keywords: Safety control, missing data

1. INTRODUCTION

Designing controllers that enforce hard state-constraints is at the core of achieving safe autonomous systems. A well-studied way of synthesizing safety controllers is via computing controlled invariant sets [Blanchini (1999)]. In these works, exact knowledge of the state is usually required for all time, whereas in the imperfect information setting, computationally expensive power set construction is necessary for a sound and complete solution [Artstein and Raković (2011); De Wulf et al. (2006)].

In this paper, we study the safety control problem under a special class of information imperfections, i.e., missing (intermittent) state measurements. Such data loss may arise due to communication package drops, sensor attacks or occlusion when a vision-based sensor is used. With these sensing flaws, we need to design controllers that use limited state feedback yet still ensure safety.

Finite-time constraint-satisfaction under missing measurements has been studied with different formalisms modeling the data loss pattern, e.g., probabilistic distributions [Amin et al. (2009)], finite languages [Rutledge et al. (2020)], all-time-data-loss [Laine et al. (2020)]. While Amin et al. (2009) only considers the satisfaction of a chance constraint, a stronger result that ensures safety for all time will require worst-case analysis as in Rutledge et al. (2020); Laine et al. (2020), which may lead to conservativeness when extended for solving infinite horizon problems. Missing data patterns over an infinite horizon can be characterized by omega automata [van Horssen et al. (2016)], and are used in controllability and observability analysis [Jungers et al. (2017)]. The safety controller synthesis in this case, however, has not been addressed to the best of our knowledge.

We consider the infinite horizon safety control problem where missing measurement patterns are specified by automata, and show that, in this setting, it is possible to solve the problem without power set construction. The key ingredients of our approach are twofold. First, we introduce the notion of string automata to capture the original automaton's underlying structure that is relevant to decision making, and to define the discrete dynamics of the controller. Second, we use the fact that a controller must be causal (i.e., at runtime, the controller does not know how many measurements are going to be missed consecutively), which we encode as safety constraints imposed on the control actions. We show that, with the above reductions, the control synthesis problem for systems with missing measurements can be solved as a full-information safety game on a product system that is constructed from the original system and the string automaton. Control synthesis on product systems is standard for discrete state systems. For dynamics with continuous state spaces, such attempts have previously been made in Danielson et al. (2019); Liu and Ozay (2019) and Li (2019) (Chapter 6). The idea, however, has not been explored for solving the synthesis problems with missing data. Moreover, unique to this paper is the usage of the string automaton and causality for the problem reduction.

2. PRELIMINARIES

Let S be a set, the set of finite and infinite sequences of elements in S are denoted by S^* and S^ω, respectively, and the set of sequences of length T is denoted by S^T. Throughout the paper, we will use bold font letters (e.g., $\mathbf{s}$) to denote finite or infinite sequences. Small letter s_t denotes the t^{th} element in sequence $\mathbf{s}$, and $\mathbf{s}_t := s_1 s_2 \dots s_t$.

Definition 1. A *deterministic automaton* is a tuple $\mathcal{A} = (S, A, s_{\text{init}}, \delta)$, where S is a finite set of states, A is a finite set of letters called the alphabet, $s_{\text{init}} \in S$ is the initial

state and $\delta \subseteq S \times A \times S$ is the transition relation. The automaton is deterministic in the sense that the initial state s_{init} is unique and $|\{s' \mid (s,a,s') \in \delta\}| \leq 1$ for all $s \in S$ and $a \in A$. Whenever $(s,a,s') \in \delta$, we write $s' = \delta(s,a)$ with a slight abuse of notation. A sequence $\mathbf{a} = a_1 a_2 \cdots \in A^\omega$ is *accepted* by an automaton $\mathcal{A}$ if and only if there exists an infinite state sequence $\mathbf{s} = s_1 s_2 s_3 \cdots \in S^\omega$, called a *run*, such that $s_1 = s_{\text{init}}$ and $s_{t+1} = \delta(s_t, a_t)$ for all $t \geq 1$. The *language* accepted by an automaton $\mathcal{A}$, denoted by $\mathcal{L}(\mathcal{A})$, is the set of all sequences accepted by $\mathcal{A}$.

We now review the safety control problem under full state information. Consider a discrete-time system $x_{t+1} = f(x_t, u_t, w_t)$ where $x_t \in X$ is the state, $u_t \in U$ is the control input, $w_t \in W$ is the (non-measured) disturbance. The problem of safety control synthesis concerns finding a state feedback controller and *the* set X_{inv} of all the initial conditions starting from where the closed-loop trajectories under all possible disturbance profiles are rendered within a give set X_{safe} of safe states. The set X_{inv} is called the maximal controlled invariant set contained by the safe set. In Bertsekas (1972), the following fixed-point iteration, starting from $X_1 = X_{\text{safe}}$, is introduced:

$$X_{n+1} = \mathbf{Proj}_x\big(\mathbf{C}(X_n)\big), \text{ for } n = 1, 2, 3 \ldots \quad (1)$$

where $\mathbf{C}(X_n) = \{(x,u) \in X_n \times U \mid \forall w \in W : f(x,u,w) \in X_n\}$. The intuition of the iteration in Eq. (1) is to remove the states at which the disturbance can force the trajectory to the unsafe set $X \setminus X_{\text{safe}}$ in finite steps. Formally, sets $Z_n = X \setminus X_n$ satisfy the following induction relation:

$$Z_{n+1} = \{x \in X \mid \forall u \in U : \exists w \in W : f(x,u,w) \in Z_n\} \cup Z_n. \quad (2)$$

Under mild conditions, X_n defined by (1) converges to X_{inv}. A control action u is *admissible* at state $x \in X_{\text{inv}}$ if $(x,u) \in \mathbf{C}(X_{\text{inv}})$. An admissible controller $C : X \to U$ (i.e., $C(x)$ is admissible at all $x \in X_{\text{inv}}$) ensures safety.

Proposition 1. (Proposition 4 in Bertsekas (1972)) If $\mathbf{C}(X_n)$ is compact for all positive integers n, $\lim_{n\to\infty} X_n = \cap_{n=1}^{\infty} X_n = X_{\text{inv}}$ and $\lim_{n\to\infty} Z_n = \cup_{n=1}^{\infty} Z_n = X \setminus X_{\text{inv}}$.

3. PROBLEM STATEMENT

To define the safety control problem under missing state measurements, consider a system $\Sigma = (X, U, W, f)$,

$$x_{t+1} = f(x_t, u_t, w_t), \quad (3)$$

$$y_t = \begin{cases} x_t & \text{if } a_t = 1 \\ \emptyset & \text{if } a_t = 0 \end{cases}, \quad (4)$$

where $x_t \in X$ is the state, $u_t \in U$ is the control input, $w_t \in W$ is the disturbance, $y_t \in Y = X \cup \{\emptyset\}$ is the state measurement and the symbol $\emptyset$ denotes a measurement loss, and $a_t \in \{0,1\}$ is a boolean indicator variable that takes zero value when the measurement is missing and one otherwise. Map $f : X \times U \times W \to X$ defines the system's evolution. Let $\mathbf{u} \in U^T$, $\mathbf{w} \in W^T$ and $\mathbf{x} \in X^{T+1}$ be the system's trajectory under $\mathbf{u}, \mathbf{w}$ from state x_1, we write $\mathbf{x} = f(x_1, \mathbf{u}, \mathbf{w})$ and $x_{\tau+1} = f_\tau(x_1, \mathbf{u}, \mathbf{w})$ for $\tau = 1, 2, \ldots, T$.

Clearly, if there is no restriction on the indicator sequences, it is allowed to have $a_t \equiv 0$ for all t, and the best strategy that a controller can take is an open-loop control sequence, which cannot effectively react to the disturbance w. In this paper, we restrict the indicator sequence $a_1 a_2 \ldots$ to be in a set $\mathcal{L}_a \subseteq \{0,1\}^\omega$ that satisfies the assumption below.

Assumption 1. Three conditions hold for the set $\mathcal{L}_a$:

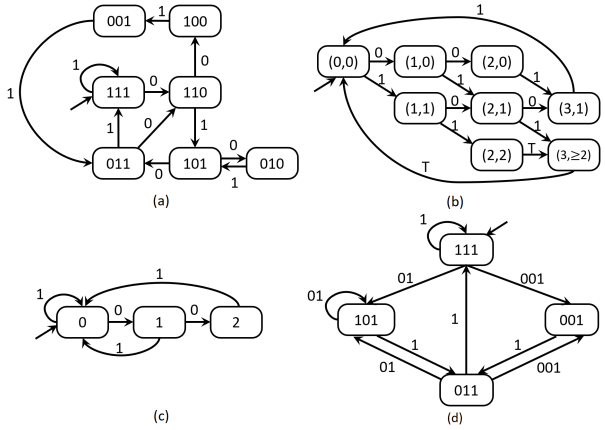

Fig. 1. Example automata: (a) $(4,2)$-firm measurement loss, (b) blocked $(4,2)$-firm measurement loss, (c) at most 2 consecutive missing measurements, (d) string automaton for $(4,2)$-firm measurement loss pattern.

i) the initial measurement is available, and the following sequence is specified by an automaton $\mathcal{A}$ whose alphabet is $\{0,1\}$, i.e., $\mathcal{L}_a = \{1\mathbf{a} \mid \mathbf{a} \in \mathcal{L}(\mathcal{A})\}$.
ii) a measurement may be available under all circumstances i.e., $\forall \mathbf{a} \in \mathcal{L}_a, t \geq 1 : \exists \mathbf{a}' \in \mathcal{L}_a : \mathbf{a}'_{t+1} = \mathbf{a}_t 1$,
iii) the number of consecutive zeros in any sequence $\mathbf{a} \in \mathcal{L}_a$ is no larger than a positive integer k.

Many missing data patterns satisfy Assumption 1, including (but not restricted to):

(a) (m,k)-firm measurement loss [Jia et al. (2005)]: the number of missing measurements within any time window of length m is no greater than k, i.e.,

$$a_1 = 1 \text{ and } \forall t \geq 1 : \textstyle\sum_{\tau=t}^{t+m-1} a_\tau \geq m - k, \quad (5)$$

(b) blocked (m,k)-firm measurement loss:

$$a_1 = 1 \text{ and } \forall l \geq 0 : \textstyle\sum_{\tau=lm+1}^{lm+m} a_\tau \geq m - k, \quad (6)$$

(c) at most k consecutive measurement loss:

$$a_1 = 1 \text{ and } \forall t \geq 1 : \textstyle\bigwedge_{\tau=t}^{t+k-1} (a_\tau = 0) \to (a_{t+k} = 1) \quad (7)$$

(d) finite union and finite intersection of any sets satisfying Assumption 1.

Example 1. Fig. 1 shows some example automata $\mathcal{A}$ and $\mathcal{L}_a = \{1\mathbf{a} \mid \mathbf{a} \in \mathcal{L}(\mathcal{A})\}$ in the above cases (a) (b) (c). The arrow with a "T" on top is a simplified notation for two transitions, one under $a = 1$ and the other under $a = 0$. In Fig. 1 (a), the string within each state can be interpreted as the last 3 digit of a_t observed. In Fig. 1 (b), each node contains a pair (z_1, z_2) where z_1 is the time steps that has passed in the current time block of length four, and z_2 is the number of available measurements observed so far in this block. In Fig. 1 (c), the number within a node represents the number of consecutive missing points observed.

The closed-loop behaviors of a system Σ are fully defined given a set $\mathcal{L}_a$ and a controller $C : Y \times (Y \times U)^* \to U$, which maps $y_1 y_2 \ldots y_t$ and $u_1 u_2 \ldots u_{t-1}$ to a control input $u_t \in U$. A trajectory $\mathbf{x} = x_1 x_2 \cdots \in X^\omega$ is generated under the controller C from an initial state x_{init} if i) $x_1 = x_{\text{init}}$, and ii) there exists $\mathbf{a} \in \mathcal{L}_a, \mathbf{w} \in W^\omega, \mathbf{u} \in U^\omega, \mathbf{y} \in Y^\omega$ s.t. for all $t \geq 1$, $x_{t+1} = f(x_t, u_t, w_t)$, where $u_t = C(\mathbf{y}_t, \mathbf{u}_{t-1})$ and y_τ, a_τ, x_τ satisfy Eq. (4) for all $1 \leq \tau \leq t$.

Now we formally define the problem studied in this paper.

Problem 1. Given a system Σ in Eq. (3), (4), a set $X_{\text{safe}} \subseteq X$ of safe states and a set $\mathcal{L}_a$ of admissible measurement loss indicator sequences that satisfy Assumption 1, find *the* set X_{win} of initial states, called the *winning set*, together with a safety controller C, such that all trajectories $\mathbf{x}$ generated under C starting from X_{win} remain in the safe set for all time, i.e., $x_t \in X_{\text{safe}}$ for all $t \geq 1$.

4. SOLUTION APPROACH

Missing measurements can be viewed as a special form of partial information. Hence the complete solution to Problem 1 can be obtained by solving a full-information game using the power set of the original state space. Such power set construction, however, is expensive and even intractable when considering a system with continuous state space. For safety control problems with imperfect information, computationally efficient methods exist but at the cost of completeness [Yang and Ozay (2020)]. For the instances studied in this paper, where $\mathcal{L}_a$ is defined by an automaton, the synthesis problem can be cast into a full-information safety game on a product system, which is constructed from the original system and a *string automaton*. The winning set and admissible controllers can be then computed by applying the fixed-point algorithm from Bertsekas (1972) to this product system. The proposed algorithm is sound, complete, and applicable to continuous-state systems.

4.1 String Automaton

We first introduce the string automaton. The purpose is to remove the states irrelevant to decision making from the state space of the automaton specifying language $\mathcal{L}_a$.

A string automaton $\overline{\mathcal{A}} = (\overline{S}, \overline{A}, s_{\text{init}}, \overline{\delta})$ is nothing but a deterministic automaton (see Definition 1), whose alphabet $\overline{A} \subseteq A^*$ is a set of finite sequences, called the strings, of letters from some other alphabet A. The only difference is that the language accepted by a string automaton $\overline{\mathcal{A}}$ is parsed over A^ω instead of $(\overline{A})^\omega$. We let $\overline{\mathcal{L}}(\overline{\mathcal{A}})$ denote the language accepted by $\overline{\mathcal{A}}$ when parsed over A^ω, and conceptually distinguish it from $\mathcal{L}(\overline{\mathcal{A}})$, which by Definition 1 is the language accepted by $\overline{\mathcal{A}}$ when parsed over $\overline{A}^\omega$. In our case where the language to be characterized consists of measurement loss indicator sequences, the string alphabet $\overline{A} \subseteq \{0,1\}^*$. Particularly, for a reason that will become clear later, we restrict $\overline{A} = \{1, 01, 001, \ldots, (0)^k 1\}$ where $(0)^k 1$ represents a sequence of k consecutive zeros followed by a one, and k is the maximum number of consecutive missing data in Assumption 1, bullet iii). We denote a string from $\overline{A}$ by $\overline{a}$.

Let $\mathcal{A}$ be the automaton that accepts a language $\mathcal{L}_a$ satisfying Assumption 1, the following algorithm constructs a string automaton $\overline{\mathcal{A}}$ that accepts the same language. In line 5 of Algorithm 1, we write $s' = \delta(s, \overline{a})$ to mean that there exists a sequence $s_1 s_2 \ldots s_{|\overline{a}|+1}$ such that $s_1 = s$, $s_{|\overline{a}|+1} = s'$ and $s_{t+1} = \delta(s_t, a_t)$ for all $1 \leq t \leq |\overline{a}|$, where $\overline{a} = a_1 a_2 \ldots a_{|\overline{a}|}$ and $|\overline{a}|$ stands for the length of string $\overline{a}$.

Example 2. Consider the language $\mathcal{L}(\mathcal{A})$ of (4,2)-firm measurement loss patterns, which is accepted by the automaton in Fig. 1 (a). The associated string automaton returned by Algorithm 1 is shown in Fig. 1 (d), where the binary sequence in each node represents a state s, the sequence on each arrow represents a letter $\overline{a}$.

Algorithm 1 $\overline{\mathcal{A}} = \textbf{StringAuto}(\mathcal{A})$

1: $Q = \{s_{\text{init}}\}$, $\overline{S} = \emptyset$, $\overline{A} = \{1, 01, \ldots, (0)^k 1\}$, $\overline{\delta} = \emptyset$
2: **while** $Q \neq \emptyset$ **do**
3: pick $s \in Q$, $Q = Q \setminus \{s\}$, $\overline{S} = \overline{S} \cup \{s\}$
4: **for** $\overline{a} \in \overline{A}$ **do**
5: **if** $s' = \delta(s, \overline{a})$ is defined **then**
6: $\overline{\delta} = \overline{\delta} \cup \{(s, \overline{a}, s')\}$, $Q = Q \cup (\{s'\} \setminus \overline{S})$
return $\overline{\mathcal{A}} = (\overline{S}, \overline{A}, s_{\text{init}}, \overline{\delta})$.

Proposition 2. Let $\mathcal{A}$ be an automaton over alphabet $\{0,1\}$ such that $\mathcal{L}(\mathcal{A})$ satisfy Assumption 1, and let $\overline{\mathcal{A}} = \textbf{StringAuto}(\mathcal{A})$, we have $\overline{\mathcal{L}}(\overline{\mathcal{A}}) = \mathcal{L}(\mathcal{A})$.

By construction, the state set $\overline{S}$ of a string automaton $\overline{\mathcal{A}} = \textbf{StringAuto}(\mathcal{A})$ is a subset of that of the original automaton $\mathcal{A}$, while the two automata characterize the same language. The significance of such reduction is that the states in $\overline{S}$ are exactly the ones that are relevant to the decision making. In fact, if $\mathcal{A}$ has just reads a sequence ending with zero at time t (i.e., $a_t = 0$), the control action at time t has to depend on the previous available measurements and could have been determined when the last available measurement was received. Moreover, it only makes sense to make a new decision when the next available measurement comes, i.e., after the automaton reads a sequence in the form of $(0)^l 1$ for some $l \leq k$. This naturally leads to the the notion of string automata constructed in Algorithm 1, whose alphabets are in form of $\overline{A} = \{1, 01, 001, \ldots, (0)^k 1\}$.

4.2 Causality of the Controller

In this part, we discuss how to use string automata for decision making. The main idea is based on a property of the controller called causality. At runtime, a safety controller does not know in advance how many consecutive missing points are coming before the next available measurement. Instead it can only plan (i.e., determine an open-loop control sequence) for the worst-case scenario where the maximum number of consecutive missing measurements allowed by Assumption 1 is reached. We shall refer to this property as the causality of the controller. This maximum number is not necessarily k in Assumption 1 iii) and also depends on the finite a_t-sequence observed by far. In particular, this maximum number $\ell(s)$ can be determined by tracking the state s of the string automaton:

$$\ell(s) := \max_{\overline{a} \in \overline{A}_s} |\overline{a}| \quad \text{where } \overline{A}_s := \{\overline{a} \mid \exists s' = \overline{\delta}(s, \overline{a})\}. \quad (8)$$

Take the string automaton in Figure 1 (d) as an example and let the current system's state be x_t and the current automaton state be $s_t = 111$. At this state, there could be at most two consecutive missing measurements in the future (i.e., $\ell(s_t) = 3$). The controller hence need to determine an open-loop control sequence $u_t u_{t+1} u_{t+2} \in U^{\ell(s_t)}$ of length three based on $y_t = x_t$ [1], otherwise the control inputs at time instants $t+1$ and $t+2$ are not defined in case of $\overline{a} = 001$. In practice, there could be

[1] Since there is no measurement noise, it is sufficient for the controller to base its decision on y_t, but not the early measurements.

only one (corresponding to $\overline{a} = 01$) or even no missing measurements at all (corresponding to $\overline{a} = 1$) before the next available measurement, but the controller will execute the first two (in case of $\overline{a} = 01$) or the first one (in case of $\overline{a} = 1$) control input in the same open-loop sequence $u_t u_{t+1} u_{t+2}$ nonetheless, because it does not know the real $\overline{a}$ until the next available measurement comes.

4.3 Product System

Since the decision is made based on both the system state x and the string automaton state s, this naturally leads to the following definition of product systems.

Definition 2. Let $\Sigma = (X, U, W, f)$ be a system and $\overline{\mathcal{A}} = (\overline{S}, \overline{A}, s_{\text{init}}, \overline{\delta})$ be a string automaton, the product system, denoted by $\Sigma \otimes \overline{\mathcal{A}}$, is a tuple $(\mathfrak{X}, \mathfrak{U}, \mathfrak{W}, \mathfrak{f})$, where

- the state set $\mathfrak{X} := X \times \overline{S}$,
- the control set $\mathfrak{U} := \bigcup_{s \in \overline{S}} U^{\ell(s)}$,
- the disturbance set $\mathfrak{W} := \bigcup_{s \in \overline{S}} \overline{A}_s \times W^{\ell(s)}$,
- the transition relation $\mathfrak{f} \subseteq \mathfrak{X} \times \mathfrak{U} \times \mathfrak{W} \times \mathfrak{X}$ is defined to be such that $\big((x,s), \mathbf{u}, (\overline{a}, \mathbf{w}), (x', s')\big) \in \mathfrak{f}$ iff i) $\mathrm{u} \in U^{\ell(s)}$, ii) $(\overline{a}, \mathbf{w}) \in \overline{A}_s \times W^{\ell(s)}$, iii) $s' = \overline{\delta}(s, \overline{a})$ and iv) $x' = f_{|\overline{a}|}(x, \mathbf{u}, \mathbf{w})$. In this case we write $(x', s') = \mathfrak{f}\big((x,s), \mathbf{u}, (\overline{a}, \mathbf{w})\big)$.

The product system defined above is essentially the dynamics of the system state x and the string automaton state s sampled at the time instants when a decision occurs, i.e., when a available measurement has just been received. The control set $\mathfrak{U}$ contains open-loop sequences. Disturbance set $\mathfrak{W}$ consists of pairs $(\overline{a}, \mathbf{w})$ that are picked by the environment, where $\mathbf{w}$ is the disturbance profile and $\overline{a}$ determines which measurements are missing. In particular, to respect the missing data pattern $\mathcal{L}_a$, the admissible set of variable $\overline{a}$ depends on the current automaton state s, and all the possible values of $\overline{a}$ that are allowed at state s is defined by $\overline{A}_s$.

A state-feedback controller $\mathfrak{C} : \mathfrak{X} \to \mathfrak{U}$ for the product system is *valid* if $\mathfrak{C}(x,s) \in U^{\ell(s)}$. A controller $C : Y \times (Y \times U)^* \to U$ for the original system can be induced by

$$C(y_1) = \mathfrak{C}(y_1, s_{\text{init}}), \tag{9}$$

$$C(y_1 y_2 \dots y_t, u_1 u_2 \dots u_{t-1}) = u'_{t-t^\star+1}, \tag{10}$$

where $u'_1 u'_2 \dots u'_{\ell(s')} := \mathfrak{C}(y_{t^\star}, s')$, $s' := \overline{\delta}(s_{\text{init}}, \overline{\mathbf{a}}_{t^\star})$, $t^\star := \max\{1 \le \tau \le t \mid y_\tau \neq \emptyset\}$ and $\overline{\mathbf{a}}_{t^\star} = a_2 \dots a_{t^\star}$ is such that $a_\tau = 1$ if $y_\tau \neq \emptyset$ and $a_\tau = 0$ if $y_\tau = \emptyset$. Roughly, the controller C first finds the time instant $t^\star$ when the last available measurement was received, find the associated indicator sequence $\overline{a}_{t^\star}$ and compute the current string automaton state by $s' = \overline{\delta}(s_{\text{init}}, \overline{\mathbf{a}}_{t^\star})$, and extract the open-loop control sequence by $\mathfrak{C}(y_{t^\star}, s')$. Finally the control action at the $(t - t^\star + 1)^{\text{th}}$ position in this control sequence is applied at the current time t. The relation between the closed-loop trajectories of the product system and the original system is established by the following proposition.

Proposition 3. Let $\mathfrak{C}$ be a valid controller for the product system and C be the induced controller for the original system. Let $x_{\text{init}} \in X, \mathbf{w} \in W^\omega$ be arbitrary and $\mathbf{a} \in \{0,1\}^\omega$ be such that $1\mathbf{a} \in \mathcal{L}_a$. Define t_i so that a_{t_i} is the i^{th} one in sequence $\mathbf{a}$. and define $\overline{\mathbf{a}} = \overline{a}_1 \overline{a}_2 \dots$ with $\overline{a}_1 = (0)^{t_1 - 1}1$ and $\overline{a}_i = (0)^{t_i - t_{i-1} - 1}1$ for $i \ge 2$. Now let $\mathbf{s} \in \overline{S}^\omega$ be the run with which $\overline{\mathcal{A}}$ accepts $\overline{\mathbf{a}}$, and let $\mathbf{x}$ be the original system's trajectory initiated at $x_1 = x_{\text{init}}$ under $\mathbf{a}, \mathbf{w}$ and the induced controller C. The trajectory of the product system under $\overline{\mathbf{a}}, \mathbf{w}$ and controller $\mathfrak{C}$ is then given by $(x_{t_1}, s_1)(x_{t_2}, s_2) \dots$.

4.4 Safety Control Synthesis on the Product System

Now we are in the position to describe our solution approach. Given an instance of Problem 1 defined by the original system Σ, a string automaton $\overline{\mathcal{A}}$ and a safe set X_{safe}, it can be converted to a standard safety problem with full information on the product system $\Sigma \otimes \overline{\mathcal{A}}$. This standard safety control problem can be solved by fixed-point algorithm defined by Eq. (1), starting from the product system safe set $\mathfrak{X}_{\text{safe}} := X_{\text{safe}} \times \overline{S}$. This leads to a sequence of sets $\mathfrak{X}_n$ in the state space of the product system. Under mild conditions, we show that $\mathfrak{X}_n$ converges to the maximal controlled invariant set $\mathfrak{X}_{\text{inv}} \subseteq \mathfrak{X}_{\text{safe}}$ on the product system.

Proposition 4. Assume that sets X_{safe}, U are compact and f is continuous, we have $\lim_{n \to \infty} \mathfrak{X}_n = \mathfrak{X}_{\text{inv}}$.

Proof. All we need to show is that $\mathbf{C}(\mathfrak{X}_n)$ is compact for all n, then Proposition 4 follows from Proposition 1.

We first prove that $\mathbf{C}(\mathfrak{X}_n)$ is compact when $\mathfrak{X}_n$ is compact. Let $X_s := \{x \mid (x,s) \in \mathfrak{X}_n\}$, set $\mathbf{C}(\mathfrak{X}_n)$ can be written as

$$\bigcup_{\substack{s \in \overline{S} \\ (s, \overline{a}, s') \in \overline{\delta}, \\ \mathbf{w} \in W^{\ell(s)}}} \bigcap \underbrace{\left\{ (x, s, \mathbf{u}) \,\middle|\, \begin{array}{l} \mathbf{u} \in U^{\ell(s)},\ x \in X_s \\ f_{|\overline{a}|}(x, \mathbf{u}, \mathbf{w}) \in X_{s'} \end{array} \right\}}_{\mathbf{C}_{s, \overline{a}, s', \mathbf{w}}(\mathfrak{X}_n)}. \tag{11}$$

Since X_s, X_{safe}, U are compact and f (hence $f_{|\overline{a}|}$) is continuous, set $\mathbf{C}_{s, \overline{a}, s', \mathbf{w}}(\mathfrak{X}_n)$ is also compact, hence so is $\mathbf{C}(\mathfrak{X}_n)$ because the union in Eq. (11) is taken over a finite set $\overline{S}$. Now it remains to show $\mathfrak{X}_n$ to be compact for all n.

1° X_{safe} is assumed compact, thus so is $\mathfrak{X}_1 = \mathfrak{X}_{\text{safe}}$.
2° Assuming that $\mathfrak{X}_n$ is compact, we know that $\mathbf{C}(\mathfrak{X}_n)$ is compact by the above argument. This implies that $\mathfrak{X}_{n+1} = \mathbf{Proj}_{(x,s)}\big(\mathbf{C}(\mathfrak{X}_n)\big)$ is also compact, which completes the induction step and hence the proof. ■

It still remains to prove that finding the maximal controlled invariant set on the product system will indeed solve Problem 1.

Proposition 5. Suppose that Assumption 1 holds. Let $\mathfrak{X}_{\text{inv}}$ be the maximal controlled invariant set of the product system contained by $\mathfrak{X}_{\text{safe}}$, under controller $\mathfrak{C}$. For all $s \in \overline{S}$, define $X_s := \{x \mid (x,s) \in \mathfrak{X}_{\text{inv}}\}$. The winning set X_{win} under the missing measurement setting is equal to $X_{s_{\text{init}}}$, whose associated controller C is induced from $\mathfrak{C}$ by Eq. (10).

Proof. We first prove soundness, i.e., $X_{s_{\text{init}}} \subseteq X_{\text{win}}$. Let $x_{\text{init}} \in X_{s_{\text{init}}}$ and $1\mathbf{a} \in \mathcal{L}_a, \mathbf{w} \in W^\omega$ be arbitrary. Define $\mathbf{x}$ be the original system's trajectory generated under $1\mathbf{a}, \mathbf{w}$ and C. Define t_i, s_i from $\mathbf{a}$ as in Proposition 3, we know that $(x_{t_1}, s_1) = (x_1, s_1) = (x_{\text{init}}, s_{\text{init}}) \in \mathfrak{X}_{\text{inv}}$. Since $\mathfrak{X}_{\text{inv}}$ is invariant under $\mathfrak{C}$, we know $(x_{t_i}, s_i) \in \mathfrak{X}_{\text{safe}}$ for all i. Thus $x_{t_i} \in X_{\text{safe}}$ for all i.

It remains to show that the intermediate states $x_\tau \in X_{\text{safe}}$ for all $t_i < \tau < t_{i+1}$. By Assumption 1 iii), we know that

$$\forall t_i < \tau < t_{i+1} : \exists s' \in \overline{S} : (s_i, (0)^{\tau - t_i - 1} 1, s') \in \overline{\delta}. \quad (12)$$

Also note that $\big(x_{t_i}, s_i, \mathfrak{C}(x_{t_i}, s_i)\big) \in \mathbf{C}(\mathfrak{X}_{\text{inv}})$, hence

$$\forall (s_i, \overline{a}, s') \in \overline{\delta} : f_{|\overline{a}|}\big(x_{t_i}, \mathfrak{C}(x_{t_i}, s_i), w_{t_i} \dots w_{t_{i+1}}\big) \in X_{s'}. \quad (13)$$

Combing Eq. (12), (13), we have

$$\forall t_i < \tau < t_{i+1} : \exists s' \in \overline{S} :$$
$$x_\tau = f_{\tau - t_i}\big(x_{t_i}, \mathfrak{C}(x_{t_i}, s_i), w_{t_i} \dots w_{t_{i+1}}\big) \in X_{s'} \quad (14)$$

Since $X_{s'} \subseteq X_{\text{safe}}$ by construction, we have $x_\tau \in X_{\text{safe}}$.

Next, we prove the completeness, i.e., $X_{\text{win}} \subseteq X_{s_{\text{init}}}$. Let $x_{\text{init}} \notin X_{s_{\text{init}}}$ be arbitrary, we will show that $x_{\text{init}} \notin X_{\text{win}}$ by contradiction. Assume for a contradiction that $x_{\text{init}} \in X_{\text{win}}$, then there exists a controller C such that all closed-loop trajectories starting from x_{init} stay in X_{safe} indefinitely. But since $x_{\text{init}} \notin X_{s_{\text{init}}}$, we know that $(x_{\text{init}}, s_{\text{init}}) \notin \mathfrak{X}_{\text{inv}}$. Hence, by Proposition 1, there exists integer N s.t. $(x_{\text{init}}, s_{\text{init}}) \in \mathfrak{Z}_N$, where $\mathfrak{Z}_n = \{Z_{s,n}\}_{s \in \overline{S}}$ is defined by applying the iteration in (2) to the product system starting from $\mathfrak{Z}_1 = \mathfrak{X} \backslash \mathfrak{X}_{\text{safe}}$ for n steps. This means either i) $(x_{\text{init}}, s_{\text{init}}) \in \mathfrak{Z}_{N-1}$, i.e., $x_{\text{init}} \in Z_{s_{\text{init}}, N}$, or ii)

$$\forall \mathbf{u} \in U^{\ell(s_{\text{init}})} : \exists (\overline{a}, \mathbf{w}) \in \overline{A}_{s_{\text{init}}} \times W^{\ell(s_{\text{init}})} :$$
$$\mathfrak{f}\big((x, s_{\text{init}}), \mathbf{u}, (\overline{a}, \mathbf{w})\big) \in \mathfrak{Z}_{N-1} \quad (15)$$
$$\Updownarrow$$
$$\forall \mathbf{u} \in U^{\ell(s_{\text{init}})} : \exists (s_{\text{init}}, \overline{a}, s') \in \overline{\delta}, \mathbf{w} \in W^{\ell(s_{\text{init}})} :$$
$$f_{|\overline{a}|}(x, \mathbf{u}, \mathbf{w}) \in Z_{s', N-1}. \quad (16)$$

Now consider control sequence $u_1 u_2 \dots u_{\ell(s_{\text{init}})}$ where $u_t = C(x_{\text{init}}(\emptyset)^{t-1})$. Clearly $u_1 u_2 \dots u_t$ will be applied sequentially by controller C if $\overline{a} = (0)^{t-1}1$ is observed. But by Eq. (16), we know that there exists $\overline{a}, s', \mathbf{w} \in W^{\ell(s_{\text{init}})}$ such that $s' = \overline{\delta}(s_{\text{init}}, \overline{a})$ and $x_{|\overline{a}|+1} = f_{|\overline{a}|}(x, u_1 u_2 \dots u_{\ell(s_{\text{init}})}, \mathbf{w}) \in Z_{s', N-1}$. Combining case i) and ii) above, we know that the closed-loop system's states under controller C can be forced into $Z_{s', N-1}$ for some s' in finite steps. Applying such argument inductively proves that the closed-loop trajectory initiated at x_{init} under C can be forced into $Z_{s'', 1} = X \setminus X_{\text{safe}}$ for some other s'' in finte steps, which contradicts that controller C ensures safety when the system starts from x_{init}. ■

Remark 1. We conclude this part by some computational remarks. First, the conditions in Proposition 4 can cover systems with discrete or continuous state spaces. For these systems, in principle, the set $\mathbf{C}(\mathfrak{X}_n)$ in Eq. (11) (and the associated finite open-loop strategy) can be computed in a very similar way as proposed in Laine et al. (2020). For linear systems with additive control actions and disturbances, the computation reduces to Minkowski sum and differences (see Example 4), and finite termination of the fixed-point iteration can be enforced at the cost of completeness (i.e., finding a inner approximation of the maximal controlled invariant set), using the technique proposed by Rungger and Tabuada (2017). For discrete-state system, the fixed-point iteration terminates in finite time.

4.5 Systems with Partially Missing State Measurements

In practice, different from the output map in Eq. (4), the controller may access part of the state x^1 perfectly for all time, while occasionally miss the rest part of the states x^2 with a pattern specified by $\mathcal{L}_a$. That is

$$y_t = \begin{cases} (x_t^1, x_t^2) & \text{if } a_t = 1 \\ (x_t^1, \emptyset) & \text{if } a_t = 0 \end{cases}. \quad (17)$$

This will in general lead to more complicated information structures that requires constructing an observer. However, our approach applies directly whenever the dynamics of x^1 is undisturbed and not affected by x^2, i.e., $x_{t+1}^1 = f^1(x_t^1, u_t)$. In this case, there is no better decision time other than the instants whenever x^2 is properly measured.

5. EXAMPLES

We illustrate our approach with two examples, one with finite state space and one with continuous state space. The goal is to show how the winning set and the admissible control actions are affected by measurement loss.

Example 3. Consider a finite state system that abstracts the car-pedestrian dynamics at an uncontrolled intersection. The state $x = (c, v, p) \in X := \{1, 2, \dots, c_{\max}\} \times \{0, 1, \dots, v_{\max}\} \times \{1, 2, \dots, p_{\max}\}$, where c and v are the position and velocity of the ego car, and p is the position of a pedestrian. The intersection is at $c = c^* < c_{\max}$ and $p = p^* < p_{\max}$. The control input $u \in U := \{-1, 0, 1\}$ is the acceleration of the car, while uncontrolled input (i.e., disturbance) $w \in W := \{0, 1\}$ is the velocity of the pedestrian. The goal is to determine u to prevent collision, i.e, enforcing $x_t \in X_{\text{safe}} := X \setminus X_{\text{unsafe}}$ for all t, where

$$X_{\text{unsafe}} := \left\{ (c, v, p) \,\middle|\, \begin{array}{l} \big((c = c^*) \wedge (p = p^*)\big) \vee \\ \big((c < c^*) \wedge (c + v > c^*) \wedge (p = p^* - 1)\big) \end{array} \right\} \quad (18)$$

The transition is defined as follows:

- $c_{t+1} = c_t$ if $c_t = c_{\max}$, and $c_{t+1} = c_t + v_t$ otherwise.
- $v_{t+1} = \max\{v_t + u_t, v_{\max}\}$.
- $p_{t+1} = p_t + w_t$, where w_t is state-dependent:

$$w_t \in \begin{cases} \{0\} & \text{if } p_t = p_{\max} \\ \{1\} & \text{if } (c_t = c^*) \wedge (v_t = 0) \wedge (p_t \in \{p^*, p^* - 1\}) \\ \{0, 1\} & \text{otherwise} \end{cases}. \quad (19)$$

The positions of the car and the pedestrian will be fixed once hitting $c_{\max}$ and $p_{\max}$ as their changes beyond the upper limit are not of our interest. To avoid deadlocks, the car will let the pedestrian pass first whenever it stops at the intersection and the pedestrian is also close enough. This corresponds to the second line in the definition of w_t, Eq. (19). The ego car's position c and speed v are known

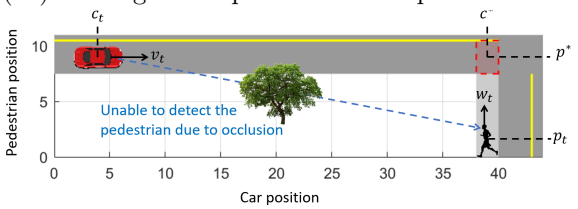

Fig. 2. Illustration of Example 3.

exactly for all time, while the pedestrian's position p might be missing because, e.g., the perception module fails to detect the pedestrian due to occlusion. The loss of p is assumed to follow a $(4, 2)$-firm pattern. Since the ego car's dynamics is neither perturbed by w nor affected by the pedestrian's position, the system falls into the category discussed in Section 4.5, and our approach is directly applicable. In this example, the winning set under missing data is identical to that under full information of x. This

is because the car can always slow down and stop at the intersection, and this open-loop strategy does not depend on the knowledge of pedestrian's position p. However, in case of measurement loss, the admissible actions at each state are more restrictive. For example, if we want the car to cross the intersection as soon as possible, the decision under missing measurements will be more conservative and tends to slow down even if the pedestrian stops moving, in which case a controller with full information will be more aggressive and accelerate. This is demonstrated in Figure 3. The simulation starts at $(c, v, p) = (5, 7, 4)$, with $c^* = 39$, $p^* = 9$. Comparing to the control decisions under $(4, 2)$-firm measurement (blue dotted), an open-loop strategy (green dashed) only ensures safety, but forbids the ego car to pass the intersection without knowing the pedestrian's position.

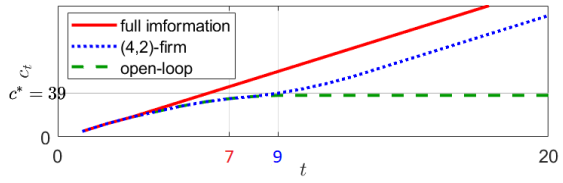

Fig. 3. The position of the ego car under $\mathbf{w} = 0(1)^\omega$. The vertical lines remark the time instants when the car crosses the intersection.

Example 4. Consider linear systems in the following form

$$\Sigma : x_{t+1} \in Ax_t + Bu_t \oplus EW \tag{20}$$

where sets U, W and X_{safe} are polytopes. In this case, the set iteration on the product system reduces to polytope operations. Let $\mathfrak{X}_{n+1} = \mathbf{Proj}_x\big(\mathbf{C}(\mathfrak{X}_n)\big)$ be the n^{th} iteration, and define $X_{s,n} := \{x \mid (x, s) \in \mathfrak{X}_n\}$ and $X_{s,n+1} := \{x \mid (x, s) \in \mathfrak{X}_{n+1}\}$ for every string automaton state $s \in \overline{S}$, we have

$$X_{s,n+1} = \left\{ x \in X_{\text{safe}} \middle| \begin{array}{l} \big(\forall \overline{a} \in \overline{A}, s' \in \overline{S} : s' = \overline{\delta}(s, \overline{a})\big): \\ A^{|\overline{a}|}x \in X_{s',n} \ominus \big(\bigoplus_{i=0}^{|\overline{a}|-1} A^i EW\big) \\ \qquad \oplus \big(-\bigoplus_{i=0}^{|\overline{a}|-1} A^i BU\big) \end{array} \right\}, \tag{21}$$

where the Minkowski sum ("$\oplus$") and difference ("$\ominus$") are implemented by MPT3 toolbox [Herceg et al. (2013)]. In this example, $U = [-1, 1]$, $W = [-0.9, 0.9]$,

$$A = \begin{bmatrix} 0.9930 & 0.03584 \\ -0.2240 & 0.9930 \end{bmatrix}, B = E = \begin{bmatrix} -0.005324 \\ 0.1205 \end{bmatrix}. \tag{22}$$

The lossy measurements follow a $(7, 4)$-firm pattern. Figure 4 shows the obtained winning sets. It can be seen that, after convergence, set X_s (blue transparent) varies slightly with s, and there is a gap between the winning set under missing measurements (black dashed line)and that under full information (red solid line).

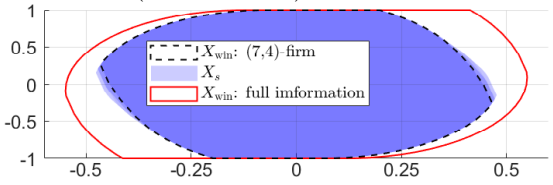

Fig. 4. Example 4: winning sets.

Acknowledgments. This work is supported by the ONR grant N00014-18-1-2501 and the NSF grant ECCS-1553873.

REFERENCES

Amin, S., Cárdenas, A.A., and Sastry, S.S. (2009). Safe and secure networked control systems under denial-of-service attacks. In *International Workshop on Hybrid Systems: Computation and Control*, 31–45. Springer.

Artstein, Z. and Raković, S.V. (2011). Set invariance under output feedback: a set-dynamics approach. *International Journal of Systems Science*, 42(4), 539–555.

Bertsekas, D. (1972). Infinite time reachability of state-space regions by using feedback control. *IEEE Transactions on Automatic Control*, 17(5), 604–613.

Blanchini, F. (1999). Set invariance in control. *Automatica*, 35(11), 1747–1767.

Danielson, C., Bridgeman, L.J., and Di Cairano, S. (2019). Necessary and sufficient conditions for constraint satisfaction in switched systems using switch-robust control invariant sets. *International Journal of Robust and Nonlinear Control*, 29(9), 2589–2602.

De Wulf, M., Doyen, L., and Raskin, J.F. (2006). A lattice theory for solving games of imperfect information. In *International Workshop on Hybrid Systems: Computation and Control*, 153–168. Springer.

Herceg, M., Kvasnica, M., Jones, C., and Morari, M. (2013). Multi-Parametric Toolbox 3.0. In *Proc. of the European Control Conference*, 502–510. Zürich, Switzerland. `http://control.ee.ethz.ch/~mpt`.

Jia, N., Song, Y.Q., and Lin, R.Z. (2005). Analysis of networked control system with packet drops governed by (m,k)-firm constraint. In *Fieldbus Systems and Their Applications*, 63–70. Elsevier.

Jungers, R.M., Kundu, A., and Heemels, W. (2017). Observability and controllability analysis of linear systems subject to data losses. *IEEE Transactions on Automatic Control*, 63(10), 3361–3376.

Laine, F., Chiu, C.Y., and Tomlin, C. (2020). Eyes-closed safety kernels: Safety for autonomous systems under loss of observability. *arXiv preprint arXiv:2005.07144*.

Li, Y. (2019). *Robustly complete temporal logic control synthesis for nonlinear systems*. Ph.D. thesis, University of Waterloo.

Liu, Z. and Ozay, N. (2019). Safety control with preview automaton. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, 1557–1564. IEEE.

Rungger, M. and Tabuada, P. (2017). Computing robust controlled invariant sets of linear systems. *IEEE Transactions on Automatic Control*, 62(7), 3665–3670.

Rutledge, K., Yong, S.Z., and Ozay, N. (2020). Finite horizon constrained control and bounded-error estimation in the presence of missing data. *Nonlinear Analysis: Hybrid Systems*, 36, 100854.

van Horssen, E.P., Behrouzian, A.B., Goswami, D., Antunes, D., Basten, T., and Heemels, W. (2016). Performance analysis and controller improvement for linear systems with (m, k)-firm data losses. In *2016 European Control Conference (ECC)*, 2571–2577. IEEE.

Yang, L. and Ozay, N. (2020). Efficient safety control synthesis with imperfect state information. In *2020 59th IEEE Conference on Decision and Control (CDC)*, 874–880. IEEE.