A Lattice Theory for Solving Games of Imperfect Information *

Martin De Wulf, Laurent Doyen**, and Jean-François Raskin

Département d'Informatique
Université Libre de Bruxelles

Abstract. In this paper, we propose a fixed point theory to solve games of imperfect information. The fixed point theory is defined on the lattice of antichains of sets of states. Contrary to the classical solution proposed by Reif [Rei84], our new solution does not involve determinization. As a consequence, it is readily applicable to classes of systems that do not admit determinization. Notable examples of such systems are timed and hybrid automata. As an application, we show that the discrete control problem for games of imperfect information defined by rectangular automata is decidable. This result extends a result by Henzinger and Kopke in [HK99].

1 Introduction

Timed and hybrid systems are dynamical systems with both discrete and continuous components. A paradigmatic example of a hybrid system is a digital control program for an analog plant environment, like a furnace or an airplane: the controller state moves discretely between control modes, and in each control mode, the plant state evolves continuously according to physical laws. A natural model for hybrid systems is the *hybrid automaton*, which represents discrete components using finite-state machines and continuous components using real-numbered variables whose evolution is governed by differential equations or differential inclusions [ACH$^+$95].

The distinction between continuous evolutions of the plant state (which is given by the real-numbered variables of a hybrid automaton) and discrete switches of the controller state (which is given by the location, or control mode, of the hybrid automaton) permits a natural formulation of the *safety control problem*: given an unsafe set U of plant states, is there a strategy to switch the controller state in real time so that the plant can be prevented from entering U? In other words, the hybrid automaton specifies a set of possible control modes, together with the plant behavior resulting from each mode, and the control problem asks for deriving a *switching strategy* between control modes that keeps the plant out of trouble.

In the literature, there are algorithms or semi-algorithms (termination is not always guaranteed) to derive such switching strategy. Those semi-algorithms usually comes in the form of symbolic fixed point computations that manipulate sets of states using a

* Supported by the FRFC project "Centre Fédéré en Vérification" funded by the Belgian National Science Fundation (FNRS) under grant nr 2.4530.02

** Research fellow supported by the Belgian National Science Foundation (FNRS).

well-suited monotonic function like the controllable predecessor operator [AHK02,MPS95]. Those algorithms make a strong hypothesis: they consider that the controller that executes the switching strategy has a *perfect information* about the state of the controlled system. Unfortunately, this is usually an unreasonable hypothesis. Indeed, when the switching strategy has to be implemented by a real hardware, the controller typically acquires information about the state of the system by reading values on sensors. Those sensors have finite precision, and so the information about the state in which the system lies is *imperfect*. Let us illustrate this. Consider a controller that monitors the temperature of a tank, and has to maintain the temperature between given bounds by switching on and off a gas burner. The temperature of the tank is the state of the continuous system to control. Assume that the temperature is sensed through a thermometer that returns an integer number and ensures a deviation bounded by one degree Celsius. So, when the sensor returns the temperature c, the controller only knows that the temperature lies in the interval $(c-1, c+1)$ degrees. We say that the sensor reading is an *observation* of the system. This observation gives an imperfect information about the state of the system.

Now, if we fix a set of possible observations of the system to control, the control problem that we want to solve is the *safety control problem with imperfect information*: "given an unsafe set U of plant states, a set of observations, is there an observation based strategy to switch the controller state in real time so that the plant can be prevented from entering U?". While it is well-known that safety games of perfect information can be won using *memoryless strategies*, it is not the case for games of imperfect information [Rei84]. In that paper, Reif studies *games of incomplete information* which are a subclass of safety games of imperfect information where the set of observations is a partition of the state space. Notice that this is not the case of our tank example since when the temperature of the water is d, the thermometer may return either $\lceil d \rceil$ or $\lfloor d \rfloor$. To win such games, memory is sometimes necessary: the controller has to remember (part of) the history of observations that he has made so far. In the finite state case, games of incomplete information can be solved algorithmically. Reif proposes an algorithm that first transforms the game of incomplete information into a game of perfect information using a kind of determinization procedure.

In this paper, we propose an alternative method to solve games of imperfect (and incomplete) information. Our method comes in the form of a fixed point (semi-)algorithm that iterates a monotone operator on the lattice of antichains of sets of states. The greatest fixed point of this operator contains exactly the information needed to determine the states from which an observation based control strategy exists and to synthesize such a strategy. We prove that our algorithm has an optimal complexity for finite state games and we identify a class of infinite state games for which the greatest fixed point of the operator is computable. Using this class of games and results from [HK99], we show that the discrete-time control problem with imperfect information is decidable for the class of rectangular automata. Strategies that win those games have some robustness properties as they can be implemented using hardware that senses its environment with finite precision.

Our fixed point method has several advantages over the algorithmic method proposed by Reif. First, as it does not require determinization, our (semi-)algorithm is readily applicable to classes of systems for which determinization is not effective: timed and hybrid automata are notable examples [AD94]. Second, we show that there are families of games on which the Reif's algorithm needs exponential time when our algorithm only needs polynomial time. Third, as our method is based on a lattice theory, abstract interpretation methods can be used to derive in a systematic way approximation algorithms [CC77].

Our paper is structured as follows. In Section 2, we recall the definition of the lattice of antichains. In Section 3, we show how to use this lattice to solve games of imperfect information. In Section 4, we give a fixed point algorithm that is EXPTIME for finite state games and we compare with the technique of Reif. Finally, in Section 5, we solve games of imperfect information for rectangular automata. Due to lack of space, the proofs of most of the theorems have been omitted and can be found in [**?**]

2 The Lattice of Antichains

First we recall what is the lattice of antichains. An *antichain* on a partially ordered set $\langle X, \leq \rangle$ is a set $X' \subseteq X$ such that for any $x_1, x_2 \in X'$ with $x_1 \neq x_2$ we have neither $x_1 \leq x_2$ nor $x_2 \leq x_1$, that is X' is a set of incomparable elements of X. We define similarly a *chain* to be a set of comparable elements of X.

Let $q, q' \in 2^{2^S}$ and define $q \sqsubseteq q'$ if and only if $\forall s \in q : \exists s' \in q' : s \subseteq s'$. This relation is a preorder but is not antisymmetric. Since we need a partial order, we construct the set $L \subseteq 2^{2^S}$ for which $\sqsubseteq$ is antisymmetric on L. The set L is the set of antichains on $\langle 2^S, \subseteq \rangle$.

We say that a set $s \subseteq S$ is *dominated* in q if and only if $\exists s' \in q : s \subset s'$. The set of dominated elements of q is denoted $\mathsf{Dom}(q)$. The *reduced form* of q is $\lceil q \rceil = q \backslash \mathsf{Dom}(q)$ and dually the *expanded* form of q is $\rceil q \lceil = q \cup \mathsf{Dom}(q)$. The set $\lceil q \rceil$ is an antichain of $\langle 2^S, \subseteq \rangle$. Observe that $\mathsf{Dom}(\lceil q \rceil) = \emptyset$, that is $\forall s, s' \in \lceil q \rceil$: if $s_1 \subseteq s_2$ then $s_1 = s_2$. The relation $\sqsubseteq$ has the useful following properties:

Lemma 1 *Let $q, q' \in 2^{2^S}$. If $q \subseteq q'$ then $q \sqsubseteq q'$.*

Lemma 2 $\forall q, q' \in 2^{2^S}, \forall q_1, q_2 \in \{q, \lceil q \rceil, \rceil q \lceil\}, \forall q_1', q_2' \in \{q', \lceil q' \rceil, \rceil q' \lceil\} : q_1 \sqsubseteq q_2$ *is equivalent to* $q_1' \sqsubseteq q_2'$.

We can now define formally L as the set $\{\lceil q \rceil \mid q \in 2^{2^S}\}$.

Lemma 3 *The relation $\sqsubseteq \subseteq L \times L$ is a partial ordering and $\langle L, \sqsubseteq \rangle$ is a partially ordered set.*

Lemma 4 *For $q, q' \in L$, the greatest lower bound of q and q' is $q \bigsqcap q' = \lceil \{ s \cap s' \mid s \in q \wedge s' \in q' \} \rceil$ and the least upper bound of q and q' is $q \bigsqcup q' = \lceil \{ s \mid s \in q \vee s \in q' \} \rceil$.*

For $Q \subseteq L$, we have $\bigsqcap Q = \lceil \{ \bigcap_{q \in Q} s_q \mid s_q \in q \} \rceil$ and $\bigsqcup Q = \lceil \{ s \mid \exists q \in Q : s \in q \} \rceil$. The least element of L is $\bot = \bigsqcap L = \emptyset$ and the greatest element of L is $\top = \bigsqcup L = \{S\}$.

Lemma 5 $\langle L, \sqsubseteq, \bigsqcup, \bigsqcap, \bot, \top \rangle$ *is a complete lattice.*

This lattice is the *lattice of antichains*.

3 Games of Imperfect Information

3.1 Definitions

Notations Given a finite sequence $\overline{a} = a_0, a_1, \ldots, a_n$, we denote by $|\overline{a}| = n + 1$ the length of $\overline{a}$, by $\overline{a}_k = a_0, \ldots, a_k$ the sequence of the first $k + 1$ elements of $\overline{a}$ (and $\overline{a}_{-1}$ is the empty sequence) and by $\mathsf{last}(\overline{a}) = a_n$ the last element of $\overline{a}$.

Definition 6 [Two-player games] A *two-player game* is a tuple $\langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow\rangle$ where S is a set of states, $S_0 \subseteq S$ is the set of *initial* states, Σ^c (resp. Σ^u) is a finite alphabet of *controllable* (resp. *uncontrollable*) *actions*, and $\rightarrow \subseteq S \times (\Sigma^c \cup \Sigma^u) \times S$ is a transition relation.

The game is turn-based and played by a controller against an environment. To initialize the game, the environment chooses a state $x \in S_0$ and the controller takes the first turn. A turn of the controller consists of choosing a controllable action σ that is enabled in the current state x. If no such action exists, the controllers loses. A turn of the environment then consists of determining a state y such that $x \xrightarrow{\sigma} y$ and of choosing an uncontrollable action u and a state z such that $y \xrightarrow{u} z$. If no enabled action u exists the environment loses. If the game continues forever, the controller wins.

For $\sigma \in \Sigma^c \cup \Sigma^u$, let $\mathsf{Enabled}(\sigma) = \{x \in S \mid \exists x' \in S : (x, \sigma, x') \in\rightarrow\}$ be the set of states in which the action σ is *enabled*, and for $s \subseteq S$ let $\mathsf{Post}_\sigma(s) = \{x' \in S \mid \exists x \in s : (x, \sigma, x') \in\rightarrow\}$ be the set of *successor states* of s by the action σ. Furthermore, given a set $\Sigma \subseteq \Sigma^c \cup \Sigma^u$, we define the notation $\mathsf{Post}_\Sigma(s)$ to mean $\bigcup_{\sigma\in\Sigma} \mathsf{Post}_\sigma(s)$.

The controller has an imperfect view of the game state space in that his/her choices are based on imprecise observations of the states.

Definition 7 [Observation set] An *observation set* of the state space S is a couple (Obs, γ) where $\gamma : \mathsf{Obs} \rightarrow 2^S$ is such that for all $x \in S$, there exists $\mathsf{obs} \in \mathsf{Obs}$ such that $x \in \gamma(\mathsf{obs})$.

An observation obs is *compatible* with a state x if $x \in \gamma(\mathsf{obs})$. When the controller observes the current state x of the game, he/she receives *one* observation compatible with x. The observation is non-deterministically chosen by the environment.

Definition 8 [Imperfect information] A two-player game $\langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow\rangle$ equipped with an observation set (Obs, γ) of its state space defines a *game of imperfect information* $\langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow, \mathsf{Obs}, \gamma\rangle$. The *size* of the game is the sum of the sizes of the transition relation $\rightarrow$ and the set Obs.

Let $G = \langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow, \mathsf{Obs}, \gamma\rangle$ be a game of imperfect information. We say that G is a *game of incomplete information* if for any $\mathsf{obs}_1, \mathsf{obs}_2 \in \mathsf{Obs}$, if $\mathsf{obs}_1 \neq \mathsf{obs}_2$ then $\gamma(\mathsf{obs}_1) \cap \gamma(\mathsf{obs}_2) = \emptyset$, that is the observations are disjoint, thus partitioning the state space. We say that G is a *game of perfect information* if $\mathsf{Obs} = S$ and γ is the identity function.

The drawback of games of incomplete information is that they are not suited for a robust modelization of sensors. Indeed, real sensors are imprecise and may return different observations for a given state.

An *observation based strategy* for a game of imperfect information $G = \langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow, \mathsf{Obs}, \gamma\rangle$ is a function $\lambda : \mathsf{Obs}^+ \rightarrow \Sigma^c$. The *outcome* of λ on G is the set $\mathsf{Outcome}_\lambda(G)$ of couples $(\overline{x}, \overline{\mathsf{obs}}) \in S^+ \times \mathsf{Obs}^+$ such that (i) $|x| = |\mathsf{obs}|$, (ii) $x_0 \in S_0$, (iii) for all $0 \le i \le |x|$, $x_i \in \gamma(\mathsf{obs}_i)$, and (iv) for all $1 \le i \le |x|$, there exists $u \in \Sigma^u$ such that $x_i \in \mathsf{Post}_u(\mathsf{Post}_{\lambda(\overline{\mathsf{obs}}_{i-1})}(\{x_{i-1}\}))$.

Definition 9 [Winning strategy] We say that an observation based strategy λ for a game G of imperfect information is *winning* if for every $(\overline{x}, \overline{\mathsf{obs}}) \in \mathsf{Outcome}_\lambda(G)$, we have $\mathsf{last}(\overline{x}) \in \mathsf{Enabled}(\lambda(\overline{\mathsf{obs}}))$.

Let us call an *history* a couple $(\overline{\mathsf{obs}}_k, \overline{\sigma}_{k-1}) \in \mathsf{Obs}^+ \times \Sigma^{c+}$ such that $\exists \overline{x} \in S^+$: $x_0 \in S_0$ and for all $0 \le i \le k$ we have $x_i \in \gamma(\mathsf{obs}_i))$ and for all $\le i < k$ we have $x_{i+1} \in \mathsf{Post}_{\Sigma^u}(\mathsf{Post}_{\sigma_i}(x_i))$.

Let us call *knowledge* after an history $(\overline{\mathsf{obs}}_k, \overline{\sigma}_{k-1})$ the function $K : \mathsf{Obs}^+ \times \Sigma^{C+} \rightarrow 2^S$ defined inductively as follows.

$$\begin{cases} K(\overline{\mathsf{obs}}_0, \overline{\sigma}_{-1}) = \gamma(\mathsf{obs}_0) \cap S_0 \\ K(\overline{\mathsf{obs}}_k, \overline{\sigma}_{k-1}) = \gamma(\mathsf{obs}_k) \cap \mathsf{Post}_{\Sigma^u}(\mathsf{Post}_{\sigma_{k-1}}(K(\overline{\mathsf{obs}}_{k-1}, \overline{\sigma}_{k-2}))) \text{ for } k > 0 \end{cases}$$

Thus, the *knowledge* after an history $(\overline{\mathsf{obs}}_k, \overline{\sigma}_{k-1})$ is the set of states the player can be sure the game is in after this history.

The *imperfect information control problem* for a class $\mathcal{C}$ of games of imperfect information is defined as follows: given a game $G \in \mathcal{C}$, determine whether there exists a winning observation based strategy for G. We define similarly the *incomplete information control problem* and the *perfect information control problem*.

Safety games We can encode the classical safety games using our winning condition. To show that, we first need some definitions. Given a game of imperfect information G we say that a set of state S_b is *final* if $\forall \sigma \in \Sigma^c \cup \Sigma^u : \mathsf{Post}_\sigma(S_b) \subseteq S_b$.

We say that a strategy λ is *safe* on a game of imperfect information G w.r.t. a final set of bad states $S_b \subseteq S$ if for every $(\overline{x}, \overline{\mathsf{obs}}) \in \mathsf{Outcome}_\lambda(G)$ we have $\mathsf{last}(\overline{x}) \notin S_b$.

The *imperfect information safety control problem* for a class $\mathcal{C}$ of games of imperfect information is defined as follows: given a two-player game $G \in \mathcal{C}$ and a final set of states S_b of G, determine whether there exists an observation based strategy λ which is safe w.r.t S_b.

Theorem 10 *The imperfect information safety control problem can be reduced to the imperfect information control problem.*

3.2 Using the Lattice of Antichains

We show how the lattice of antichains that we have introduced in Section 2 can be used to solve games of imperfect information by iterating a predecessor operator.

Controllable predecessors For $q \in L$, define the set of *controllable predecessors* of q as follows:

$$\begin{aligned}\mathsf{CPre}(q) = \lceil \{ s \subseteq S \mid \exists \sigma \in \Sigma^c : s \subseteq \mathsf{Enabled}(\sigma) \\ \wedge \forall \mathsf{obs} \in \mathsf{Obs}, \exists s' \in q : \mathsf{Post}_{\Sigma^u}(\mathsf{Post}_\sigma(s)) \cap \gamma(\mathsf{obs}) \subseteq s' \} \rceil\end{aligned}$$

Let us consider an antichain $q = \{s'_0, s'_1, \dots\}$. A set s belongs to $\mathsf{CPre}(q)$ iff (i) there is a controllable action σ that is enabled in each state of s, (ii) when the controller plays σ, any observation compatible with the next state reached by the game (after the environment has played) suffices to determine in which set s'_i of q that next state lies [1], and (iii) s is maximal .

Lemma 11 *The operator* $\mathsf{CPre} : L \to L$ *is monotone for the partial ordering* $\sqsubseteq$.

Remark The controllable predecessor operator is also *monotone w.r.t. the set of observations* in the following sense: given a two-player game G, let CPre_1 (resp. CPre_2) be the operator defined on the set of observations $(\mathsf{Obs}_1, \gamma_1)$ (resp. $(\mathsf{Obs}_2, \gamma_2)$). If $\{\gamma_2(\mathsf{obs}) \mid \mathsf{obs} \in \mathsf{Obs}_2\} \sqsubseteq \{\gamma_1(\mathsf{obs}) \mid \mathsf{obs} \in \mathsf{Obs}_1\}$, then for any $q \in L$ we have $\mathsf{CPre}_1(q) \sqsubseteq \mathsf{CPre}_2(q)$. That corresponds to the informal statement that it is easier to control a system with more precise observations.

Theorem 12 *Let* $G = \langle S, S_0, \Sigma^c, \Sigma^u, \to, \mathsf{Obs}, \gamma \rangle$ *be a game of imperfect information. There exists an observation based strategy winning on* G *if and only if*

$$\{S_0 \cap \gamma(\mathsf{obs}) \mid \mathsf{obs} \in \mathsf{Obs}\} \sqsubseteq \bigsqcup \{q \mid q = \mathsf{CPre}(q)\}. \tag{1}$$

Before proving this theorem, we give some intuition. We denote by Win the set $\bigsqcup\{q \mid q = \mathsf{CPre}(q)\}$ which is the greatest fixed point of CPre. Condition (1) states that any observation of the initial state x_0 suffices to determine in which set s of Win the game has been started. Since Win is a fixed point of the controllable predecessor operator, we know that in each set s of Win we have a controllable action that can be played by the controller in every state $x \in s$ such that (i) the state z reached after the move of the environment lies in one of the sets s' of Win whatever the environment does and, such that (ii) the set s' can be determined using any observation compatible with z. Following this, there exists a winning strategy if Condition (1) holds. The other direction of the theorem is a direct consequence of Tarski's Theorem.

Proof of Theorem 12.

First, we give an effective construction of a winning strategy for G, in the form of a finite automaton. For $q \in L$ and $\sigma \in \Sigma^c$, let $\phi(q, \sigma) = \lceil \{ s \in S \mid s \subseteq \mathsf{Enabled}(\sigma)$ and $\forall \mathsf{obs} \in \mathsf{Obs}, \exists s' \in q : \mathsf{Post}_{\Sigma^u}(\mathsf{Post}_\sigma(s)) \cap \gamma(\mathsf{obs}) \subseteq s' \} \rceil$ be the set of controllable predecessors of q for the action σ. From the greatest fixed point Win of CPre, we define the finite state automaton $A = \langle Q, q_0, \mathcal{L}, \delta \rangle$ where

- $Q = \mathsf{Win} \cup \{q_0\}$ where $q_0 \notin \mathsf{Win}$,

[1] The quantification over obs is universal since for observations that are incompatible with the new state, the condition holds trivially.

- q_0 is the initial state,
- $\mathcal{L} : Q \backslash \{q_0\} \to \Sigma^c$ is a labeling of the states. For each $s \in \mathsf{Win}$, we choose $\sigma \in \Sigma^c$ such that $s \in \phi(\mathsf{Win}, \sigma)$ and we fix $\mathcal{L}(s) = \sigma$ (such a σ exists since Win is a fixed point of CPre).
- $\delta : Q \times \mathsf{Obs} \to Q$ is a transition function.
  - For each $\mathsf{obs} \in \mathsf{Obs}$, choose $s \in \mathsf{Win}$ such that $S_0 \cap \gamma(\mathsf{obs}) \subseteq s$ and fix $\delta(q_0, \mathsf{obs}) = s$;
  - For each $s \in \mathsf{Win}$ and $\mathsf{obs} \in \mathsf{Obs}$, choose $s' \in \mathsf{Win}$ such that $\mathsf{Post}_{\Sigma^u}(\mathsf{Post}_\sigma(s)) \cap \gamma(\mathsf{obs}) \subseteq s'$ where $\sigma = \mathcal{L}(s)$ and fix $\delta(s, \mathsf{obs}) = s'$.

  Such sets s, s' exist by condition (1).

In this automaton, states are labelled with actions and transitions are labelled with observations. Intuitively, a state s of A corresponds to the minimal knowledge that is sufficient to control the system and the label $\mathcal{L}(s)$ is a winning move the controller can play having this knowledge. The next state s' is determined by the observation obs according to the transition relation.

Let $\hat{\delta} : Q \times \mathsf{Obs}^+ \to Q$ be an extension of the transition function δ on words defined recursively by $\hat{\delta}(s, \mathsf{obs}) = \delta(s, \mathsf{obs})$ and $\hat{\delta}(s, \overline{\mathsf{obs}}.\mathsf{obs}) = \delta(\hat{\delta}(s, \overline{\mathsf{obs}}), \mathsf{obs})$.

The strategy defined by A is $\lambda : \mathsf{Obs}^+ \to \Sigma^c$ such that $\lambda(\overline{\mathsf{obs}}) = \mathcal{L}(s)$ if $\hat{\delta}(q_0, \overline{\mathsf{obs}}) = s$. If for some $\overline{\mathsf{obs}}$ there is no s such that $\hat{\delta}(q_0, \overline{\mathsf{obs}}) = s$, then the sequence of observations $\overline{\mathsf{obs}}$ is impossible. In this case, we can set $\lambda(\overline{\mathsf{obs}})$ to any value.

Now we proceed with the proof of the theorem.

- If (1) holds. We show that the strategy λ defined by A is such that for any $(\overline{x}, \overline{\mathsf{obs}}) \in \mathsf{Outcome}_\lambda(G)$, we have (i) $\mathsf{last}(\overline{x}) \in \hat{\delta}(q_0, \overline{\mathsf{obs}})$ and (ii) $\mathsf{last}(\overline{x}) \in \mathsf{Enabled}(\lambda(\overline{\mathsf{obs}}))$ (thus λ is winning) . We show this by induction on the length of $\overline{x}$ and $\overline{\mathsf{obs}}$.
  1. $|\overline{x}| = 1$. We have $\overline{x} = x_0$ and $\overline{\mathsf{obs}} = \mathsf{obs}_0$ with $x_0 \in S_0$ and $x_0 \in \gamma(\mathsf{obs}_0)$. Let $s = \hat{\delta}(q_0, \mathsf{obs}_0)$ and $\sigma = \mathcal{L}(s) = \lambda(\mathsf{obs}_0)$. By construction of A, we have $S_0 \cap \gamma(\mathsf{obs}_0) \subseteq s$ and $s \in \mathsf{Win}$.
     As $x_0 \in s$ and Win is a fixed point of CPre, we have (i) $\mathsf{last}(\overline{x}) \in \hat{\delta}(q_0, \mathsf{obs}_0)$ and (ii) $x_0 \in \mathsf{Enabled}(\lambda(\mathsf{obs}_0))$.
  2. $|\overline{x}| > 0$. We have $\overline{x} = x_0, x_1, \ldots, x_k$ and $\overline{\mathsf{obs}} = \mathsf{obs}_0, \mathsf{obs}_1, \ldots, \mathsf{obs}_k$ with $x_k \in \gamma(\mathsf{obs}_k)$. Let $s_{k-1} = \hat{\delta}(q_0, \overline{\mathsf{obs}}_{k-1})$ and $\sigma = \mathcal{L}(s_{k-1}) = \lambda(\overline{\mathsf{obs}}_{k-1})$.
     By the induction hypothesis, we have $x_{k-1} \in s_{k-1}$. For $\mathsf{obs} = \mathsf{obs}_k$, let $s_k = \delta(s_{k-1}, \mathsf{obs})$. By construction of A, we have $s_k \in \mathsf{Win}$ and $\mathsf{Post}_{\Sigma^u}(\mathsf{Post}_\sigma(s_{k-1})) \cap \gamma(\mathsf{obs}) \subseteq s_k$. Therefore, we have $x_k \in s_k$ and by definition of $\mathcal{L}$, we have $s_k \subseteq \mathsf{Enabled}(\sigma')$ where $\sigma' = \mathcal{L}(s_k) = \lambda(\overline{\mathsf{obs}}_k)$. This yields (i) $\mathsf{last}(\overline{x}) \in \hat{\delta}(q_0, \overline{\mathsf{obs}})$ and (ii) $x_k \in \mathsf{Enabled}(\lambda(\overline{\mathsf{obs}}_k))$.
- If λ is an observation based strategy that is winning on G. We must show that (1) holds. Let $V_\lambda \subseteq 2^S \times \mathsf{Obs}^+$ be the smallest set (w.r.t. to $\subseteq$) such that:
  - $(S_0 \cap \gamma(\mathsf{obs}), \mathsf{obs}) \in V_\lambda$ for every $\mathsf{obs} \in \mathsf{Obs}$, and
  - if $(s, \overline{\mathsf{obs}}) \in V_\lambda$ then $(\mathsf{Post}_{\Sigma^u}(\mathsf{Post}_{\lambda(\overline{\mathsf{obs}})}(s)) \cap \gamma(\mathsf{obs}), \overline{\mathsf{obs}}.\mathsf{obs}) \in V_\lambda$ for every $\mathsf{obs} \in \mathsf{Obs}$.

  Let $W_\lambda = \{s \mid (s, \overline{\mathsf{obs}}) \in V_\lambda\}$. Let us show that $W_\lambda \sqsubseteq \mathsf{CPre}(W_\lambda)$. By Lemma 1, it suffices to show that $W_\lambda \subseteq \rceil\mathsf{CPre}(W_\lambda)\lceil$. Let $(s, \overline{\mathsf{obs}}) \in V_\lambda$ with $\overline{\mathsf{obs}} = \mathsf{obs}_0, \mathsf{obs}_1, \ldots, \mathsf{obs}_k$ and let us show that $s \in \mathsf{CPre}(W_\lambda)$.

Algorithm 1: Algorithm for CPre.

Data : A game of imperfect information $G = \langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow, \mathsf{Obs}, \gamma \rangle$ and a set $q \in L$.

Result : The set $Z = \mathsf{CPre}(q)$.

begin

1 $Z \leftarrow \emptyset$;

2 $\mathsf{Wait} \leftarrow \{S\}$;

3 **while** $\mathsf{Wait} \neq \emptyset$ **do**

4 Pick $s \in \mathsf{Wait}$ of maximal cardinality ;

5 $\mathsf{Wait} \leftarrow \mathsf{Wait}$;

6 **if** *for some* $\sigma \in \Sigma^c$ *we have :* *(1)* $s \subseteq \mathsf{Enabled}(\sigma)$ *and* *(2) for all* $\mathsf{obs} \in \mathsf{Obs}$, *there exists* $s' \in q$ *such that* $\mathsf{Post}_{\Sigma^u}(\mathsf{Post}_\sigma(s)) \cap \gamma(\mathsf{obs}) \subseteq s'$ **then**

7 $Z \leftarrow Z \cup \{s\}$;

else

8 $\mathsf{Wait} \leftarrow \mathsf{Wait} \cup \{s' \mid s' \in \mathsf{Children}(s) \wedge \forall s'' \in Z \cup \mathsf{Wait} : s' \not\subseteq s''\}$;

9 **return** Z;

end

By definition of V_λ, there exist $s_0, s_1, \ldots, s_k$ such that $s_0 = S_0 \cap \gamma(\mathsf{obs}_0)$, $s_k = s$, and for each $1 \leq i \leq k$: $s_i = \mathsf{Post}_{\Sigma^u}(\mathsf{Post}_{\sigma_i}(s_{i-1})) \cap \gamma(\mathsf{obs}_i)$ with $\sigma_i = \lambda(\mathsf{obs}_0\mathsf{obs}_1 \ldots \mathsf{obs}_{i-1})$. For any sequence of states $\overline{x} = x_0, x_1, \ldots, x_k$ with $x_i \in s_i$ and $(\overline{x}_k, \overline{\mathsf{obs}}_k) \in \mathsf{Outcome}_\lambda(G)$, since λ is winning on G, we have $x_k \in \mathsf{Enabled}(\lambda(\overline{\mathsf{obs}}))$ and thus $s \subseteq \mathsf{Enabled}(\lambda(\overline{\mathsf{obs}}))$. Also we have $\mathsf{Post}_{\Sigma^u}(\mathsf{Post}_{\lambda(\overline{\mathsf{obs}})}(s)) \cap \gamma(\mathsf{obs}) \in W_\lambda$ for every $\mathsf{obs} \in \mathsf{Obs}$ by construction of V_λ. This entails that $s \in \rceil \mathsf{CPre}(W_\lambda) \lceil$, showing that $W_\lambda \sqsubseteq \mathsf{CPre}(W_\lambda)$, that is CPre is extensive at W_λ and by the Tarski's fixed point Theorem $W_\lambda \sqsubseteq \mathsf{Win}$. The conclusion follows since $\{S_0 \cap \gamma(\mathsf{obs}) \mid \mathsf{obs} \in \mathsf{Obs}\} \subseteq W_\lambda$.

■

4 Games with Finite State Space

In this section we show that computing the greatest fixed point of CPre for finite state games can be done in EXPTIME. We also compare our algorithm based on the lattice of antichains with the classical technique of [Rei84].

4.1 Fixed Point Algorithm

To compute the greatest fixed point of CPre, we iterate CPre from S using Algorithm 1. This algorithm constructs systematically subsets of S and checks at line 6 whether they belong to $\mathsf{CPre}(q)$. This is done by treating all subsets of size i before the subsets of size $i-1$, so we avoid to treat the subsets of the already included subsets and the result is in reduced form. Therefore, Algorithm 1 uses the following operator $\mathsf{Children}(s) = \{s \backslash \{x\} \mid x \in s\}$ which returns the subsets of s of cardinality $|s|-1$.

Lemma 13 *Algorithm 1 computes* CPre *in EXPTIME in the size of the game.*

Lemma 14 *An ascending (or descending) chain in* $\langle L, \sqsubseteq, \bigsqcup, \bigsqcap, \bot, \top \rangle$ *has at most* $2^n + 1$ *elements where* $n = |S|$.

Theorem 15 *The imperfect information control problem is EXPTIME-complete.*

Proof. We first prove the upper bound. From Lemma 14 and since CPre is monotone, we reach the greatest fixed point Win after at most $O(2^n)$ iterations of CPre. From Lemma 13 computing CPre can be done in EXPTIME. The conclusion follows. For the lower bound, since we solve a more general problem than Reif [Rei84], we have the EXPTIME-hardness. ■

4.2 Example

Consider the two-player game G_1 on Fig. 1 with state space $S = \{1, 1', 2, 2', 3, 3', \mathrm{Bad}\}$, initial state $S_0 = \{2, 3\}$, actions $\Sigma^c = \{a, b\}$ and $\Sigma^u = \{u\}$. The observation set is $\mathsf{Obs} = \{\mathsf{obs}_1, \mathsf{obs}_2\}$ with $\gamma(\mathsf{obs}_1) = \{1, 1', 2, 2', \mathrm{Bad}\}$ and $\gamma(\mathsf{obs}_2) = \{1, 1', 3, 3'\}$.

For the controller, the goal is to avoid state Bad in which there is no controllable action. So the controller must play an a in state 1 and 3 and a b in state 2. However the controller cannot distinguish 1 from 2 using only the current observation. Thus, to discriminate those states, the controller has to rely on its memory of the past observations.

We show below the iterations of the fixed point algorithm and the construction of the strategy. The fixed point computation starts from $\top = \{S\}$. Each set is paired with an action that can be played in all the states of that set:

$$
\begin{aligned}
S_1 &= \mathsf{CPre}(\{S\}) = \{\{1, 2, 3\}_a\} \\
S_2 &= \mathsf{CPre}(S_1) \quad = \{\{2\}_b, \{1, 3\}_a\} \\
S_3 &= \mathsf{CPre}(S_2) \quad = \{\{1\}_a, \{2\}_b, \{3\}_a\} \\
S_4 &= \mathsf{CPre}(S_3) \quad = S_3
\end{aligned}
$$

Since $S_4 = S_3$, we have $\mathsf{Win} = S_3 = \{\{1\}, \{2\}, \{3\}\}$. The existence of a winning strategy is established by condition (1) of Theorem 12 since the sets $S_0 \cap \gamma(\mathsf{obs}_1) = \{2\}$ and $S_0 \cap \gamma(\mathsf{obs}_2) = \{3\}$ are dominated in Win.

From the fixed point, using the construction given in the proof of Theorem 12, we construct the automaton of Fig. 2 which encodes a winning strategy. Indeed, when the game starts the control is either in state 2 if the given observation is obs_1 or in state 3 if the given observation is obs_2. In the first case, the controller plays b and in the second case, it plays a. Then the game lies in state 1. According to the strategy automaton, the controller plays an a and receives a new observation that allows it to determine if the game lies now in state 2 (obs_1) or in state 3 (obs_2). From there, the controller can clearly iterate this strategy.

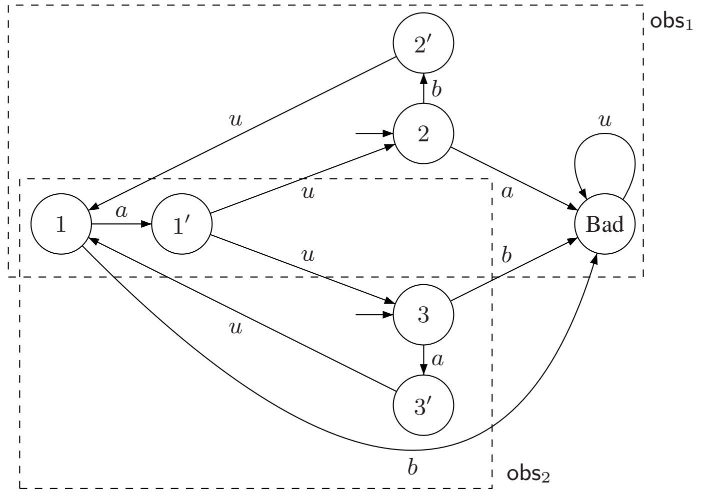

Fig. 1. A two-player game G_1 with observation set $\{\mathsf{obs}_1, \mathsf{obs}_2\}$.

4.3 Comparison with the classical technique of [Rei84]

In [Rei84] the author gives an algorithm to transform a game of incomplete information G into a game G' of perfect information on the histories of G.

The idea can be expressed as follows : given a game of incomplete information $G = \langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow_1, \mathsf{Obs}, \gamma\rangle$ define a two-player game $G'' = \langle S', S'_0, \Sigma^c, \{\varepsilon\}, \rightarrow_2\rangle$ as follows: S' is the set of knowledges $K(\overline{\mathsf{obs}}_k, \overline{\sigma}_{k-1})$ such that $(\overline{\mathsf{obs}}_k, \overline{\sigma}_{k-1})$ is an history of G. S'_0 is the set of knowledges $\{K(\mathsf{obs}_0) | \gamma(\mathsf{obs}_0) \cap S_0 \neq \emptyset\}$. Finally the transition relation $\rightarrow_2$ is defined as follows: $K(\overline{\mathsf{obs}}_k, \overline{\sigma}_{k-1}) \xrightarrow{\sigma_k}_2 K(\overline{\mathsf{obs}}_{k+1}, \overline{\sigma}_k)$ and $s \xrightarrow{\varepsilon}_2 s$ for all $s \in S'$. To obtain the final game of perfect information G', equip G'' with the set of observation (S', γ_I) where γ_I is the identity function. Solving the resulting game

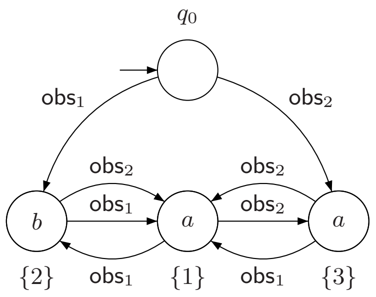

Fig. 2. A finite state automaton A defining a winning strategy for G_1.

of perfect information G' requires linear time in the size of S' but there exist games of incomplete information G requiring the construction of a game of perfect information of size exponentially larger than the size of G.

As our algorithm does not require this determinization, it is easy to find families of games where our method is exponentially faster than Reif's algorithm. This is formalized in the next theorem.

Theorem 16 *There exist finite state games of incomplete information for which the algorithm of [Rei84] requires an exponential time where our algorithm needs only polynomial time.*

5 Control with imperfect information of rectangular automata

In this section, we introduce the notion of infinite games with finite stable quotient. We use this notion to show that the *discrete control problem for games of imperfect information defined by rectangular automata* is decidable. This result extends the results in [HK99].

5.1 Games with Finite R-stable quotient

Here we drop the assumption that S is finite and we consider the case where there exists a finite quotient of S over which the game is *stable*. We obtain a general decidability result for games of imperfect information with finite stable quotients.

Let $R = \{r_1, \dots, r_l\}$ be a finite partition of S. A set $s \subseteq S$ is *R-definable* if $s = \bigcup_{r \in Z} r$ for some $Z \subseteq R$. We say that an antichain $q \in L$ is R-definable if for every $s \in q$, s is R-definable.

Definition 17 [R-stable] A game of imperfect information $\langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow, \mathsf{Obs}, \gamma \rangle$ is *R-stable* if for every $\sigma \in \Sigma^c$ the following conditions hold: (i) $\mathsf{Enabled}(\sigma)$ is R-definable; (ii) for every $r \in R$, $\mathsf{Post}_{\Sigma^u}(\mathsf{Post}_\sigma(r))$ is R-definable; (iii) for any $r, r' \in R$, if for some $x \in r$ and $u \in \Sigma^u$ we have $\mathsf{Post}_u(\mathsf{Post}_\sigma)(\{x\}) \cap r' \neq \emptyset$ then for any $x \in r$, there exists $u \in \Sigma^u$ such that $\mathsf{Post}_{\Sigma^u}(\mathsf{Post}_\sigma)(\{x\}) \cap r' \neq \emptyset$; (iv) furthermore, for every $\mathsf{obs} \in \mathsf{Obs}$, $\gamma(\mathsf{obs})$ is R-definable.

The next lemma states properties of R-stable games of imperfect information. They are useful for the proof of the next theorem.

Lemma 18 *Let $G = \langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow, \mathsf{Obs}, \gamma \rangle$ be a R-stable game of imperfect information. Let $s, s', s'' \subseteq S$ and $r \in R$ such that (i) s' and s'' are R-definable and (ii) $s \cap r \neq \emptyset$. If there exists $\sigma \in \Sigma^c$ such that (iii) $s \subseteq \mathsf{Enabled}(\sigma)$ and (iv) $\mathsf{Post}_{\Sigma^u}(\mathsf{Post}_\sigma(s)) \cap s' \subseteq s''$ then (v) $r \subseteq \mathsf{Enabled}(\sigma)$ and (vi) $\mathsf{Post}_{\Sigma^u}(\mathsf{Post}_\sigma(s \cup r)) \cap s' \subseteq s''$.*

Theorem 19 *Let $G = \langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow, \mathsf{Obs}, \gamma \rangle$ be a R-stable game of imperfect information. The greatest fixed point of* CPre *is a R-definable antichain and is computable.*

Proof. We show that for any R-definable antichain $q \in L$, the antichain $\mathsf{CPre}(q)$ is also R-definable. Let $s \in \mathsf{CPre}(q)$. For any $r \in R$ such that $s \cap r \neq \emptyset$, we have by Lemma 18 that $s \cup r \in \mathsf{CPre}(q)$. Since $s \subseteq s \cup r$, we must have $s = s \cup r$. This shows that s is R-definable. The number of R-definable antichains is finite, and so, using Tarski's theorem, we can compute the greatest fixed point of CPre in a finite number of iterations. ■

5.2 Rectangular automata

We first recall the definition of rectangular automata and we define their associated game semantics. We recall a result of [HK99] that establishes the existence of a finite bisimulation quotient for this game semantics.

Let $X = \{x_1, \ldots, x_n\}$ be a set of real-valued variables. A *rectangular inequality* over X is a formula of the form $x_i \sim c$, where c is an integer constant, and $\sim$ is one of the following: $<, \leq, >, \geq$. A *rectangular predicate* over X is a conjunction of rectangular inequalities. The set of all rectangular predicates over X is denoted $\mathsf{Rect}(X)$. The rectangular predicate ϕ defines the set of vectors $[\![\phi]\!] = \{y \in \mathbb{R}^n | \phi[X := y] \text{ is true}\}$. For $1 \leq i \leq n$, let $[\![\phi]\!]_i$ be the projection on variable x_i of the set $[\![\phi]\!]$. A set of the form $[\![\phi]\!]$, where ϕ is a rectangular predicate, is called a *rectangle*. Given a nonnegative integer $m \in \mathbb{N}$, the rectangular predicate ϕ and the rectangle $[\![\phi]\!]$ are m*-bounded* if $|c| \leq m$ for every conjunct $x_i \sim c$ of ϕ. Let us denote $\mathsf{Rect}_m(X)$ the set of m-bounded rectangular predicate on X.

Definition 20 [Rectangular automaton] A *rectangular automaton* H is a tuple $\langle \mathsf{Loc}, \mathsf{Lab}, \mathsf{Edg}, X, \mathsf{Init}, \mathsf{Inv}, \mathsf{Flow}, \mathsf{Jump}\rangle$ where:

- $\mathsf{Loc} = \{\ell_1, \ldots, \ell_m\}$ is a finite set of *locations*;
- Lab is a finite set of *labels*;
- $\mathsf{Edg} \subseteq \mathsf{Loc} \times \mathsf{Lab} \times \mathsf{Loc}$ is a finite set of *edges*;
- $X = \{x_1, \ldots, x_n\}$ is a finite set of *variables*;
- $\mathsf{Init} : \mathsf{Loc} \to \mathsf{Rect}(X)$ gives the *initial condition* $\mathsf{Init}(\ell)$ of location ℓ. The automaton can start in ℓ with an initial valuation v lying in $[\![\mathsf{Init}(\ell)]\!]$;
- $\mathsf{Inv} : \mathsf{Loc} \to \mathsf{Rect}(X)$ gives the *invariant condition* $\mathsf{Inv}(\ell)$ of location ℓ. The automaton can stay in ℓ as long as the values of its variables lie in $[\![\mathsf{Inv}(\ell)]\!]$;
- $\mathsf{Flow} : \mathsf{Loc} \to \mathsf{Rect}(\dot{X})$ governs the evolution of the variables in each location.
- Jump maps each edge $e \in \mathsf{Edg}$ to a predicate $\mathsf{Jump}(e)$ of the form $\phi \wedge \phi' \wedge \bigwedge_{i \notin \mathsf{Update}(e)} (x'_i = x_i)$, where $\phi \in \mathsf{Rect}(X)$ and $\phi' \in \mathsf{Rect}(X')$ and $\mathsf{Update}(e) \subseteq \{1, \ldots, n\}$. The variables in X' refer to the updated values of the variables after the edge has been traversed. Each variable x_i with $i \in \mathsf{Update}(e)$ is updated nondeterministically to an arbitrary new value in the interval $[\![\phi']\!]_i$.

A rectangular automaton is m-bounded if all its rectangular constraints are m-bounded.

Definition 21 [Nondecreasing and bounded variables] Let H be a rectangular automaton, and let $i \in \{1, \ldots, n\}$. The variable x_i of H is *nondecreasing* if for every control mode $\ell \in \mathsf{Loc}$, the invariant interval $[\![\mathsf{Inv}(\ell)]\!]_i$ and the flow interval $[\![\mathsf{Flow}(\ell)]\!]_i$ are subsets of the nonnegative reals. The variable x_i is *bounded* if for every control mode $\ell \in \mathsf{Loc}$, the invariant interval $[\![\mathsf{Inv}(\ell)]\!]_i$ is a bounded set. The automaton H has *nondecreasing* (resp. *bounded*; *nondecreasing or bounded*) *variables* if all n variables of H are nondecreasing (resp. bounded; either nondecreasing or bounded).

In the sequel, all the rectangular automata that we consider are assumed to be with nondecreasing or bounded variables.

We now associate a game semantics to each rectangular automaton.

Definition 22 [Discrete game semantics of rectangular automata] The game *semantics* of a rectangular automaton $H = \langle \mathsf{Loc}, \mathsf{Lab}, \mathsf{Edg}, X, \mathsf{Init}, \mathsf{Inv}, \mathsf{Flow}, \mathsf{Jump}\rangle$ is the game $[\![H]\!]= \langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow\rangle$ where $S = \mathsf{Loc} \times \mathbb{R}^n$ is the *state space* (with $n = |X|$), $S_0 = \{(\ell, v) \in S \mid v \in [\![\mathsf{Init}(\ell)]\!]\}$ is the *initial space*, $\Sigma^c = \mathsf{Lab}$, $\Sigma^u = \{1\}$ and $\rightarrow$ contains all the tuples $((\ell, v), \sigma, (\ell', v'))$ such that:

- either there exists $e = (\ell, \sigma, \ell') \in \mathsf{Edg}$ such that $(v, v') \in [\![\mathsf{Jump}(e)]\!]$,
- or $\ell = \ell'$ and $\sigma = 1$ and there exists a continuously differentiable function $f : [0, 1] \rightarrow [\![\mathsf{Inv}(\ell)]\!]$ such that $f(0) = v$, $f(1) = v'$ and for all $t \in (0, 1)$: $\dot{f}(t) \in [\![\mathsf{Flow}(\ell)]\!]$.

Games constructed from rectangular automata are played as follows. The game is started in a location ℓ with a valuation v for the continuous variables such that $v \in [\![\mathsf{Init}(\ell)]\!]$. At each round, the controller decides to take one of the enabled edges if one exists. Then the environment updates the continuous variables by letting time elapse for 1 time unit as specified by the (nondeterministic) flow predicates. A new round is started from there. As for the games that we have considered previously, the goal of the controller is to avoid to reach states where he does not have an enabled transition to propose.

The next definition recalls the notion of bisimulation.

Definition 23 [Bisimulation] A *simulation* on the game $G = \langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow\rangle$ is a binary relation $\sim$ on the state set S such that $s_1 \sim s_2$ implies that $\forall \sigma \in \Sigma^c \cup \Sigma^u$, if $s_1 \xrightarrow{\sigma} s_1'$ then there exists s_2' such that $s_2 \xrightarrow{\sigma} s_2'$ and $s_1' \sim s_2'$. Such a relation is called a *bisimulation* if it is symmetric.

We consider the following equivalence relation between states of rectangular automata.

Definition 24 Given the game semantics $[\![H]\!]= \langle S, S_0, \mathsf{Lab}, \{1\}, \rightarrow\rangle$ of a m-bounded rectangular automaton H, define the equivalence relation $\approx_m$ on S by $(\ell, v) \approx_m (\ell', v')$ iff $\ell = \ell'$ and for all $1 \leq i \leq n$ either $\lfloor v_i \rfloor = \lfloor v_i' \rfloor$ and $\lceil v_i \rceil = \lceil v_i' \rceil$ or both v_i and v_i' are greater than m. Let us call $R_{\approx_m}$ the set of equivalence classes of $\approx_m$ on S.

The next lemma states that the number of equivalence classes for this relation is finite for any rectangular automata.

Lemma 25 *[HK99] Let H be a m-bounded rectangular automaton. The equivalence relation $\approx_m$ is the largest bisimulation of the game semantics $[\![H]\!]$.*

5.3 Control of Rectangular Automata with imperfect information

We are now in position to extend the result of [HK99] to the case of imperfect information.

Given $H = \langle \mathsf{Loc}, \mathsf{Lab}, \mathsf{Edg}, X, \mathsf{Init}, \mathsf{Inv}, \mathsf{Flow}, \mathsf{Jump}\rangle$, a m-bounded rectangular automaton, we say that the observation set (Obs, γ) is m-bounded if for each $\mathsf{obs} \in \mathsf{Obs}$, $\gamma(\mathsf{obs})$ is definable as a finite union of sets of the form $\{(l, v) \mid v \in g\}$ where g is m-bounded rectangle.

Theorem 26 *For any m-bounded rectangular automaton H with game semantics $[\![H]\!] = \langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow\rangle$, for any m-bounded observation set (Obs, γ), the game of imperfect information $\langle S, S_0, \Sigma^c, \Sigma^u, \rightarrow, \mathsf{Obs}, \gamma\rangle$ is $R_{\approx_m}$-stable.*

As corollary of Theorem 19 and Theorem 26, we have that:

Corollary 1. *The discrete control problem for games of imperfect information defined by m bounded rectangular automata and m-bounded observation sets is decidable (in 2EXPTIME).*

So far, we do not have a hardness result but we conjecture that the problem is 2EXPTIME-complete. Now, let us illustrate the discrete control problem for games of imperfect information defined by rectangular automata on an example.

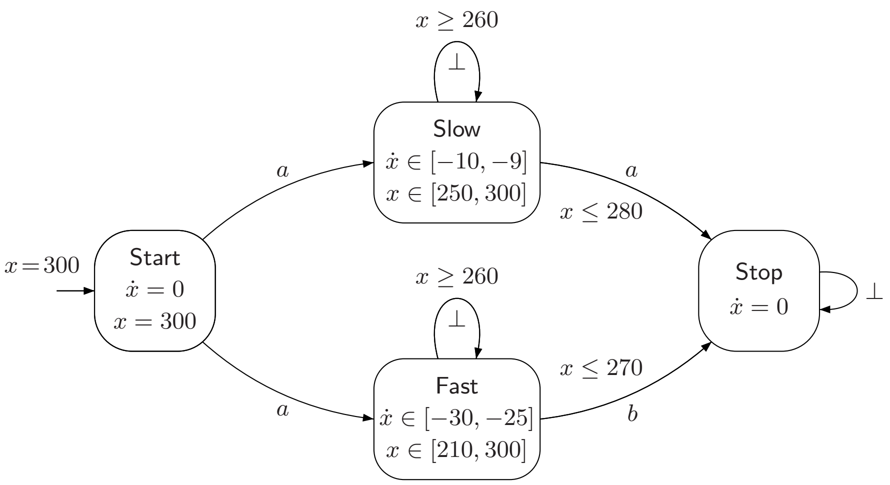

Fig. 3. A rectangular automaton modeling a cooling system.

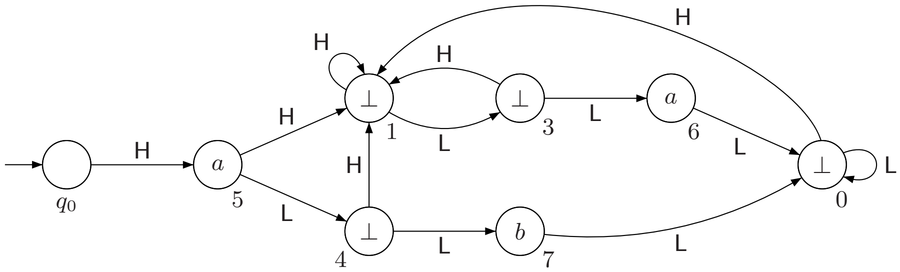

Fig. 4. A finite state automaton defining a winning strategy for the cooling system.

Example We have implemented our fixed point algorithm using HYTECH and its script language [HHWT95]. We illustrate the use of the algorithm on a simple example. Fig. 3 shows a rectangular automaton with four locations and one continuous variable x.

In this example, the game models a cooling system that controls the temperature x. When requested to start, the system begins to cool down. There are two modes of cooling, either fast or slow, among which the environment chooses. The controller can only observe the system through two observations: H with $\gamma(\mathsf{H}) = \{(\ell, x) \mid x \geq 280\}$ and L with $\gamma(\mathsf{L}) = \{(\ell, x) \mid x \leq 285\}$. Thus, only the continuous variable x can be observed imperfectly, not the modes. Depending on the mode however, the timing and action to stop the system are different. In the slow mode, the controller has to issue an action a when the temperature is below 280. In the fast mode, the controller has to issue an action b when the temperature is below 270.

The controller must use its memory of the past observations to make the correct action in time. If the first two observations are H, H then the controller knows that the mode is Slow. If the first two observations are H, L then the controller knows that the mode is Fast.

The greatest fixed point, given below, allows the computation of the deterministic strategy depicted in Fig. 4. The whole process has been automated in HYTECH. The correspondence between state numbers in the figure and states of the fixed point is the following:

- State $0 \equiv (\mathsf{Stop}, x = 0), (\mathsf{Slow}, 295 < x \leq 300)$
- State $1 \equiv (\mathsf{Slow}, 270 \leq x \leq 300)$
- (Not depicted) State $2 \equiv (\mathsf{Slow}, 295 < x \leq 300), (\mathsf{Fast}, 290 \leq x \leq 300)$
- State $3 \equiv (\mathsf{Slow}, 260 \leq x \leq 289), (\mathsf{Slow}, 295 < x \leq 300)$
- State $4 \equiv (\mathsf{Slow}, 295 < x \leq 300), (\mathsf{Fast}, 260 \leq x \leq 295)$
- State $5 \equiv (\mathsf{Start}, x = 300)$
- State $6 \equiv (\mathsf{Slow}, 250 \leq x \leq 280)$
- State $7 \equiv (\mathsf{Fast}, 210 \leq x \leq 270)$

As before, the strategy associates an action to each set of the fixed point and the observations give the next state of the strategy.

References

[ACH+95] R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[AHK02] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49:672–713, 2002.

[CC77] Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL*, pages 238–252, 1977.

[HHWT95] 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 1019, pages 41–71. Springer-Verlag, 1995.

[HK99] T.A. Henzinger and P.W. Kopke. Discrete-time control for rectangular hybrid automata. *Theoretical Computer Science*, 221:369–392, 1999.

[MPS95] O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *STACS'95*, volume 900 of *LNCS*, pages 229–242. Springer, 1995.

[Rei84] John H. Reif. The complexity of two-player games of incomplete information. *Journal of Computer and System Sciences*, 29(2):274–301, 1984.