

Centre Fédéré en Vérification

Technical Report number 2007.88

Controller Synthesis using Lattice Theory (invited tutorial)

Jean-François Raskin



This work was partially supported by a FRFC grant: 2.4530.02

<http://www.ulb.ac.be/di/ssd/cfv>

Controller Synthesis using Lattice Theory

Jean-François Raskin
CS, Université Libre de Bruxelles, U.L.B.

September 8, 2007

1 Elements of lattice theory

Let S be a set. As usual, we note 2^S for the set of subsets of S , \emptyset for the empty set, $S \times S$ for the set of pairs of elements of S , and we write $S_1 \subseteq S_2$ when the set S_1 is included in the set S_2 .

A *partial-order* over S is a relation $\sqsubseteq \subseteq S \times S$ which is (i) *reflexive*, i.e. $\forall s \in S \cdot s \sqsubseteq s$; (ii) *transitive*, i.e. $\forall s_1, s_2, s_3 \in S \cdot s_1 \sqsubseteq s_2 \wedge s_2 \sqsubseteq s_3 \rightarrow s_1 \sqsubseteq s_3$; (iii) *antisymmetric*, i.e. $\forall s_1, s_2 \in S \cdot s_1 \sqsubseteq s_2 \wedge s_2 \sqsubseteq s_1 \rightarrow s_1 = s_2$. We note (S, \sqsubseteq) , when S is partially ordered by \sqsubseteq .

Example 1 Let $S = \{s_1, s_2, s_3\}$, $(2^S, \subseteq)$ is a partially ordered set.

Let (S, \sqsubseteq) be a partially ordered set, let $s_1, s_2 \in S$, and let $T \subseteq S$. When $s_1 \sqsubseteq s_2$, we say that s_1 is *below* s_2 and s_2 is *above* s_1 . We say that s_1 is *comparable* to s_2 whenever $s_1 \sqsubseteq s_2$ or $s_2 \sqsubseteq s_1$, there are *incomparable* otherwise. T is a *chain* if any two elements of T are comparable. T is an *antichain* if any two elements of T are incomparable. s (not necessarily in T) is a *lower-bound* for T if s is below any element of T , i.e. $\forall s' \in T \cdot s \sqsubseteq s'$. An element s (not necessarily in T) is a *upper-bound* for T if s is above any element of T , i.e. $\forall s' \in T \cdot s' \sqsubseteq s$. A lower bound for T is the *greatest lower bound* (glb) for T iff it is above any lower bound for T , it is denoted $\sqcap T$ if it exists. An upper bound for T is the *least upper bound* (lub) for T iff it is below any upper bound for T , it is denoted $\sqcup T$ if it exists. A element $s \in T$ is a *minimal* (respectively *maximal*) element of T if no other element of T is below (respectively above) s . $\text{Min}(T)$ denotes the set of minimal elements of T and $\text{Max}(T)$ denotes the set of maximal elements of T . If $\text{Min}(T)$ is the singleton $\{s\}$ then s is called the *least* element of T . If $\text{Max}(T)$ is the singleton $\{s\}$ then s is called the *greatest* element of T . We write Max_{\sqsubseteq} and Min_{\sqsubseteq} if we need to make clear the underlying order.

Example 2 Let us consider $S = \{s_1, s_2, s_3\}$. In the partially ordered set $(2^S, \subseteq)$, $\{s_1, s_3\}$ is below $\{s_1, s_2, s_3\}$ as $\{s_1, s_3\} \subseteq \{s_1, s_2, s_3\}$, $\text{Max}(\{\emptyset, \{s_1\}, \{s_2, s_3\}\}) = \{\{s_1\}, \{s_2, s_3\}\}$, $\text{Max}(2^S) = \{S\}$, $\text{Min}(2^S) = \{\emptyset\}$, S is the greatest element of 2^S , and \emptyset is the least element of 2^S . $\{\{s_1\}, \{s_1, s_3\}\}$ is a chain in 2^S and $\{\{s_1\}, \{s_2, s_3\}\}$ is an antichain.

A partially ordered set (S, \sqsubseteq) is a *complete partial order*, cpo for short, iff every chain of S has a lub in S .

Example 3 Let us consider \mathcal{I} the set of intervals of reals included in $[0, 1]$. Let $\mathcal{I}^{\setminus 1}$ be the set of intervals of reals included in $[0, 1)$. (\mathcal{I}, \subseteq) and $(\mathcal{I}^{\setminus 1}, \subseteq)$ are partially ordered sets. It is easy to show that (\mathcal{I}, \subseteq) is a complete partial order. But $(\mathcal{I}^{\setminus 1}, \subseteq)$ is not a complete partial order. Indeed, let us consider the set of intervals $\{I_0, I_1, \dots, I_n, \dots\}$ where I_i is the interval $[0, 1 - \frac{1}{i+2})$. Clearly, this set of intervals is a chain but as the interval $[0, 1]$ is not in $\mathcal{I}^{\setminus 1}$ there is no lub for this chain of intervals in $\mathcal{I}^{\setminus 1}$.

A complete partial order (S, \sqsubseteq) is a *complete lattice* if every subset of S has a lub in S . As a direct consequence, every subset of S has also an glb in S . Indeed, the $\sqcap T = \sqcup \{s \mid s \text{ is a lower bound of } T\}$. So, in a complete lattice every subset of elements has a lub and a glb. The lub of \emptyset is the least element of the set and the glb of the entire set is the greatest element of the set, those elements exist. We note $\langle S, \sqsubseteq, \sqcup, \sqcap, \perp, \top \rangle$ for the complete lattice with carrier set S , partial order \sqsubseteq , least upper-bound operator \sqcup , greatest lower-bound operator \sqcap , least element \perp , greatest element \top .

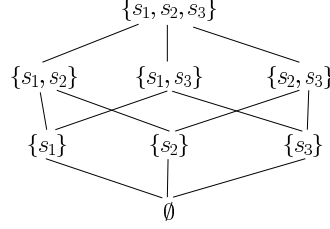


Figure 1: The powerset lattice.

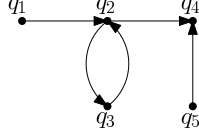


Figure 2: A transition system.

Example 4 Let S be any set, $\langle 2^S, \subseteq, \cup, \cap, \emptyset, S \rangle$ is a complete lattice. This complete lattice is called the powerset lattice of S . The powerset lattice of $S = \{s_1, s_2, s_3\}$ is depicted in Fig. 1.

A function $f : S \rightarrow S$ over a partially ordered set (S, \sqsubseteq) is *monotone* iff $\forall s_1, s_2 \in S \cdot s_1 \sqsubseteq s_2 \rightarrow f(s_1) \sqsubseteq f(s_2)$.

Example 5 A transition system is a tuple (Q, q_{init}, Δ) where Q is the set of states, $q_{init} \in Q$ is the initial state, and $\Delta \subseteq Q \times Q$ is the transition relation. Fig. 2 depicts a transition system whose nodes are the set of states $Q = \{q_1, q_2, q_3, q_4, q_5\}$, node q_1 is the initial state, and the transition relation is depicted by arrows, i.e. an arrow from a state q to a state q' denotes the fact that $(q, q') \in \Delta$. We say that q' is a successor of q when $(q, q') \in \Delta$. We say that a state q' is reachable from a state q if there exists a finite sequence of states $q_0 q_1 \dots q_n$ such that $q = q_0$, $q' = q_n$, and for all $0 \leq i < n$, $(q_i, q_{i+1}) \in \Delta$. The set of reachable states of (Q, q_{init}, Δ) is the set of states q that are reachable from q_{init} .

Let us consider the partially ordered set $(2^Q, \subseteq)$, i.e. the set of sets of states of the transition system ordered by set inclusion, and let us define the function $\text{post} : 2^Q \rightarrow 2^Q$ as $\text{post}(T) = \{q' \mid \exists q \in T \cdot \Delta(q, q')\}$. $\text{post}(T)$ is the set of states that are successors of a state in T in one step. This function is monotone over 2^Q .

For a function $f : S \rightarrow S$ over a partially ordered set (S, \sqsubseteq) , $s \in S$ is a fixed point iff $f(s) = s$. The set of fixed points of f , noted $\text{fx}(f) = \{s \mid f(s) = s\}$. In every complete partial order, every monotone function f has a least fixed point, noted $\text{lfp}(f)$ which is equal to $\sqcap \text{fx}(f)$. On a complete lattice, every monotone function f has also a greatest fixed point, noted $\text{gfp}(f)$ which is equal to $\sqcup \text{fx}(f)$.

Example 6 Let us consider again the transition system depicted in Fig. 2 and the function post that we have defined in the previous example. The set of states $\{q_2, q_3, q_4\}$ is a fixed point for the function post . Now, let us consider the function $\text{lpost} : 2^Q \rightarrow 2^Q$ which is defined, for any $T \subseteq S$, as: $\text{lpost}(T) = \{q_1\} \cup T \cup \text{post}(T)$, that is $\text{lpost}(T)$ returns the initial states together with the states that are reachable in 0 or 1 step from a state in T . The set $\{q_1, q_2, q_3, q_4\}$ is a fixed point of the function lpost and furthermore it is the least one. Note that this set is exactly the set of reachable states in the transition system.

As the last example shows, least fixed points of monotone functions are interesting objects. We now defined some additional notions necessary to define iterative scheme to evaluate them. A function $f : S \rightarrow S$ over a complete partially ordered set (S, \sqsubseteq) is *continuous* iff it is monotone and for all nonempty chains $T \subseteq S$, $f(\sqcup T) = \sqcup f(T)$. Clearly for finite sets, any monotone function is also continuous. Let $f : S \rightarrow S$, we define $f^0 = f$, and for any $i > 0$: $f^i = f \circ f^{i-1}$. The following theorem states that the least fixed point of a continuous function over a complete partial order corresponds to the limit of a simple iteration scheme due to Kleene:

Theorem 1 Let (S, \sqsubseteq) be a cpo and $f : S \rightarrow S$ be a continuous function. Then $\text{lfp}(f) = \sqcup \{f^i(\perp) \mid i \in \mathbb{N}\}$.

This theorem suggests an iteration scheme to compute the least fixed point of a function: iterate the function from the least element of the set until stabilization. This sequence is a chain which converges to the least fixed point. In any finite set, this scheme gives an algorithm which always terminates and computes the least fixed point of the function.

Example 7 Let us consider again the transition system depicted in Fig. 2, the function lpost , and apply theorem 1 to compute the least fixed point of lpost . First, remember that $\perp = \emptyset$ here, so $\text{lpost}^0(\perp) = \text{lpost}(\emptyset) = \{q_1\}$, $\text{lpost}^1(\perp) = \text{lpost}(\{q_1\}) = \{q_1, q_2\}$, $\text{lpost}^2(\perp) = \text{lpost}(\{q_1, q_2\}) = \{q_1, q_2, q_3, q_4\}$, $\text{lpost}^3(\perp) = \text{lpost}(\{q_1, q_2, q_3, q_4\}) = \{q_1, q_2, q_3, q_4\}$ which is equal to $\text{lpost}^2(\perp)$, and the iteration have reached the least fixed point of lpost which is exactly the set of states reachable from q_1 .

We can also formulate an iterative scheme to evaluate greatest fixed point that starts from the greatest element of the set.

Theorem 2 Let (S, \sqsubseteq) be a complete lattice and $f : S \rightarrow S$ be a continuous function. Then $\text{gfp}(f) = \bigcap \{f^i(\top) \mid i \in \mathbb{N}\}$.

2 Controller synthesis and games on graphs

Computer programs are more and more frequently used as controllers in embedded systems. In safety critical applications like avionics, plant control, medical instruments, etc., a high degree of reliability is necessary. To design reliable systems, methods that are based on mathematics and logic are necessary. A large research effort has been put on defining adequate models and analysis methods for embedded systems. Several international conferences on mathematical methods to reason about the correctness of embedded systems are organized each year, see for example [HT06, AB06]. Even if the verification of complex embedded systems is still a challenge, researchers are now trying to synthesize digital controllers instead of first constructing them and afterwards proving their correctness.

Framework Games playing is a powerful metaphor to think about the interaction between a controller and the environment to control. Typically, the controller must maintain the system into a set of good configurations no matter how the environment behaves. To develop algorithms for controller synthesis, we use a mathematical model known as *two-player game structures* [AHK02]. A two-player game structure $(Q, q_0, \Gamma_1, \Gamma_2, \delta)$ consists of a state space Q , an initial state q_0 , two sets of moves Γ_1 and Γ_2 and a transition function $\delta : Q \times \Gamma_1 \times \Gamma_2 \rightarrow Q$. Given a set of states $y \subseteq Q$ and an move $\gamma_1 \in \Gamma_1$ for Player 1, we note $\text{post}_{\gamma_1}(y)$ the set $\{q' \mid \exists q \in y \cdot \exists \gamma_2 \in \Gamma_2 \cdot q' = \delta(q, \gamma_1, \gamma_2)\}$, that is the set of states that can be reached from y when Player 1 chooses move γ_1 .

Games are played on two-player game structures as follows. The game starts in the initial state q_0 . In that state, each player makes a choice of move, i.e. Player 1 chooses a move $\gamma_1 \in \Gamma_1$ and Player 2 chooses a move $\gamma_2 \in \Gamma_2$, then the game evolves to state $q_1 = \delta(q_0, \gamma_1, \gamma_2)$. From q_1 , a new round is played and so on. As there is no end to that game, the result, called a *play*, is an infinite sequence of states $\rho = q_0 q_1 \dots q_n \dots$. The winner is determined by a set of winning plays $W \subseteq Q^\omega$, i.e. Player 1 is winning if $\rho \in W$ and Player 2 is winning if $\rho \in Q^\omega \setminus W$. Players are playing according to *strategies*. A strategy for Player 1 is a function $\lambda_1 : Q^* \rightarrow \Gamma_1$ that maps an prefix of a play to a choice of move for Player 1, symmetrically a Player 2 strategy is a function $\lambda_2 : Q^* \rightarrow \Gamma_2$. We say that a play $\rho = q_0 q_1 \dots q_n \dots$ is played according to strategies λ_1 and λ_2 if for all $i \geq 0$, $q_{i+1} = \delta(q_i, \lambda_1(q_0 q_1 \dots q_i), \lambda_2(q_0 q_1 \dots q_i))$. We say that ρ is the outcome of strategies λ_1 and λ_2 , and we note it $\text{Outcome}(\lambda_1, \lambda_2)$. Finally, we say that a strategy for Player 1 λ_1 is winning for objective W if for all strategies λ_2 of Player 2, we have that $\text{Outcome}(\lambda_1, \lambda_2) \in W$.

The *strategy synthesis problem* is defined as follows. Given a two-player game structure $(Q, q_0, \Gamma_1, \Gamma_2, \delta)$, given a winning objective $W \subseteq Q^\omega$ for Player 1, decide if there exists a strategy λ_1 for Player 1 such that for all strategies λ_2 of Player 2, $\text{Outcome}(\lambda_1, \lambda_2) \in W$. If such strategy exists, construct an effective representation of it. When the state space Q is finite and when the objective $W \subseteq Q^\omega$ is omega-regular, there are algorithms to solve this problem [dAHM01, dAH00].

Iterative algorithms There are elegant lattice theoretic solutions to the strategy synthesis problem, we give a summary of the main ingredients here. To keep the exposition easy, we show here how to solve finite state safety games: games where the state space Q is finite and the winning objective is defined by a subset of states $\text{Good} \subseteq Q$ that Player 1 want to stay in, i.e. $W = \{q_0q_1 \dots q_n \dots \in Q^\omega \mid \forall i \geq 0 \cdot q_i \in \text{Good}\}$. If Player 1 can win such a game, we say that Player 1 has a strategy λ_1 to ensure the safety objective Good

To check for the existence of a winning strategy for Player 1 in a safety game, we consider the complete lattice $(2^Q, \subseteq, \cup, \cap, \emptyset, Q)$ and a function $\text{Cpre}_1 : 2^Q \rightarrow 2^Q$ called the *Player 1 controllable predecessor operator*. Intuitively, this function given a set of states $x \subseteq Q$ returns the set of states $y \subseteq Q$ from which Player 1 can force the next state of the game to be in x . According to that intuition, the function is defined as follows:

$$\text{Cpre}_1(x) = \{q \in Q \mid \exists \gamma_1 \in \Gamma_1 \cdot \forall \gamma_2 \in \Gamma_2 : \delta(q, \gamma_1, \gamma_2) \in x\}$$

So, a state q is controllable for the set x if Player 1 has a choice of move such that, no matter what is the choice of move of Player 2, the next state by δ is in x .

To solve safety games with the Cpre_1 operator, we make the following observation. A set of states $x \subseteq Q$ is a set of winning states for Player 1 if $x \subseteq \text{Good}$ and Player 1 can force x from any state of x . Such a set is a fixed point for the function $f(x) = \text{Good} \cap \text{Cpre}_1(x)$. Clearly, we are interested by the largest such fixed point. This is formalized in the next theorem:

Theorem 3 *Given a two-player game structure $(Q, q_0, \Gamma_1, \Gamma_2, \delta)$, a set of states $\text{Good} \subseteq Q$, Player 1 has a strategy λ_1 to ensure the safety objective Good iff $q_0 \in \bigcup \{x \mid x = \text{Good} \cap \text{Cpre}_1(x)\}$.*

To evaluate this greatest fixed point, we can use the theorem 2. If we observe the iterations computed during this evaluation, we can see that it computes the sets of states from which Player 1 can ensure to stay within Good for 0 steps (i.e. Good), 1 steps (i.e. $\text{Good} \cap \text{Cpre}_1(\text{Good})$), etc.

3 Controller synthesis with imperfect information

The algorithms for controller synthesis that we have reviewed in Sect. 2 make a strong hypothesis: they consider that the controller has a *perfect information* about the state of the system. Unfortunately, this is usually an unreasonable hypothesis. Indeed, when the control 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*.

To model imperfect information, we fix a set of observations for the system states. The control problem that we want to solve is the *safety control problem with imperfect information*: “given a set of good states Good , a set of observations, is there an observation based strategy so that the system can be prevented from entering $Q \setminus \text{Good}$?”. We review here a fixed point based method to solve games of imperfect information, details can be found in [CDHR06, WDR06].

Framework A *two-player game structure with imperfect information* is a tuple $(Q, q_0, \gamma_1, \gamma_2, \delta, \mathcal{O})$ where $Q, q_0, \gamma_1, \gamma_2$, and δ are as for plain two-player game structures, and \mathcal{O} is a partition $\{O_1, O_2, \dots, O_n\}$ of Q into n observations. Games with imperfect information are played as follows. The game starts in state q_0 and is played in rounds. At each round, Player 1 receives the observation $O \in \mathcal{O}$ that contains the current state q , i.e. O s.t. $q \in O$, while Player 2 has perfect information on the current state of the game¹. Each player makes a choice of move, i.e. $\gamma_1 \in \Gamma_1$ and $\gamma_2 \in \Gamma_2$, and then the game evolves to the state $\delta(q, \gamma_1, \gamma_2)$. From this state, a new round is started.

As Player 1 only knows the observations of the sequence of states traversed during the game, he has to apply so-called observation based strategies. The *observation sequence* associated to a sequence of states $q_0q_1 \dots q_n$, noted $\mathcal{O}(q_0q_1 \dots q_n)$, is equal to $O_{f(0)}O_{f(1)} \dots O_{f(n)}$ such that for all $0 \leq i \leq n$, $q_i \in O_{f(i)}$. A *observation based strategy* for Player 1 is a function $\lambda_1^o : \mathcal{O}^* \rightarrow \Gamma_1$ that maps any prefix of a sequence of observations to a choice of move for Player 1. A strategy for player 2 is as before, i.e. a function $\lambda_2 : Q^* \rightarrow \Gamma_2$. The winner in a game with

¹For a discussion about this asymmetry, the interested reader is referred to [CDHR06].

imperfect information is determined by a set of winning sequences of observations $W^o \subseteq \mathcal{O}^\omega$, called an *observable objective*. We say that a play $\rho = q_0q_1 \dots q_n \dots$ is played according to strategies λ_1^o and λ_2 if for all $i \geq 0$, $q_{i+1} = \delta(q_i, \lambda_1^o(\mathcal{O}(q_0q_1 \dots q_i)), \lambda_2(q_0q_1 \dots q_i))$, ρ is called the *outcome* of the strategies λ_1^o and λ_2 , Player 1 is winning if $\mathcal{O}(\rho) \in W^o$

The *strategy synthesis problem with imperfect information* is defined as follows. Given a two-player game structure with observations $(Q, q_0, \Gamma_1, \Gamma_2, \delta, \mathcal{O})$, given an observable winning condition $W^o \subseteq \mathcal{O}^\omega$ for Player 1, decide if there exists an observation based strategy λ_1^o for Player 1 such that for all strategies λ_2 of Player 2, $\mathcal{O}(\text{Outcome}(\lambda_1^o, \lambda_2)) \in W^o$. If such strategy exists, construct an effective representation of it. When the state space Q is finite and when the objective $W^o \subseteq \mathcal{O}^\omega$ is omega-regular, there are algorithms to solve this problem [CDHR06]. We review here the solution for safety objective, i.e. we are given a set of good observation $\text{Good}^o \subseteq \mathcal{O}$, and we want to check that Player 1 has an observation based strategy against any strategy of Player 2 to stay within states that have observations in Good . If Player 1 can win such a game, we say that Player 1 has an observation based strategy to ensure the safety objective Good .

Iterative algorithm Before presenting the algorithm, we give more intuition about games of imperfect information. To better understand how Player 1 can win such games, we characterize the evolution of the *knowledge* of Player 1 during a game. The knowledge of Player 1 is the set of states in which the game can be according to what Player 1 has observed so far. Clearly, the smaller the set is, the better the knowledge is. We formalize this notion as follows. Let $G = (Q, q_0, \Gamma_1, \Gamma_2, \delta, \mathcal{O})$, let $h = O_0\gamma_0 O_1\gamma_1 \dots O_n$ be a sequence of observations and moves for Player 1 in G , h is called an *history*. We associate to the history h a set of states, noted $K(G, h)$, inductively as follows. *Base case*. If h is empty then $K(G, h) = \{q_0\}$. *Inductive case*. If $h = O_0\gamma_0 \dots O_{n-1}\gamma_{n-1} O_n$ and $\mathcal{K} = K(G, O_0\gamma_0 O_1\gamma_1 \dots O_{n-1})$ then $K(G, h) = \text{post}_{\gamma_{n-1}}(\mathcal{K}) \cap O_n$.

For games of perfect information, we have shown how to elegantly solve safety games by solving a fixed point equation constructed from the so-called controllable predecessor operator Cpre_1 . Remember that the operator Cpre_1 operates over sets of states of the game structure and that $\text{Cpre}_1(x)$ returns the set of states from which Player 1 can force the next state to be in x . In games of imperfect information, Player 1 does not know in which states the game is but only knows that the current state is in a set of possible states (the knowledge of Player 1), so we have to construct an operator that works on set of sets of states (sets of knowledges). Let $Y = \{y_1, \dots, y_n\} \subseteq 2^Q$ be a set of sets of states (knowledges) from which we know that Player 1 can force the game to stay for n steps within Good . We are interested to compute the set of knowledges $Y' = \{y'_1, \dots, y'_m\}$ from which Player 1 can force Y in one step. Note also that if the knowledge y is sufficient for Player 1 to win then clearly any knowledge $y' \subseteq y$ is also winning (Player 1 is clearly in a better situation). It is why we define below the operator for controllable predecessors not on sets of sets of states but on antichains of (maximal) sets of states.

We note $\mathcal{Y}(Q)$ for the set of antichains of knowledges over the set of states Q , i.e. antichains of sets of states. We order two antichains of knowledges as follows: let $Y = \{y_1, y_2, \dots, y_n\}$ and $Y' = \{y'_1, y'_2, \dots, y'_m\}$, $Y \preceq Y'$ iff $\forall y \in Y \cdot \exists y' \in Y' \cdot y \subseteq y'$. The partially ordered set $(\mathcal{Y}(Q), \preceq)$ forms a complete lattice with: (i) upper-bound operator \vee defined as follows: let $Y, Y' \in \mathcal{Y}(Q)$ then $Y \vee Y' = \text{Max}_{\subseteq}(Y \cup Y')$, where $\text{Max}_{\subseteq}(Y \cup Y')$ returns the maximal elements of $Y \cup Y'$ for set inclusion; (ii) lower-bound operator \wedge defined as follows: let $Y, Y' \in \mathcal{Y}(Q)$ then $Y \wedge Y' = \text{Max}_{\subseteq}(\{y \cap y' \mid y \in Y \wedge y' \in Y'\})$; (iii) minimal element \emptyset ; (iv) maximal element $\{Q\}$.

Given an antichain of knowledges $Y \in \mathcal{Y}(Q)$, we define the controllable predecessor operator for the game of imperfect information $G = (Q, q_0, \Gamma_1, \Gamma_2, \delta, \mathcal{O})$ as follows:

$$\text{Cpre}_1^o(Y) = \text{Max}_{\subseteq}(\{y' \subseteq Q \mid \exists \gamma_1 \in \Gamma_1 \cdot \forall O \in \mathcal{O} \cdot \exists y \in Y : \text{post}_{\gamma_1}(y') \cap O \subseteq y\})$$

Intuitively, given a set of knowledges Y , a knowledge $y \subseteq Q$ is controllable for Player 1 if there is a choice of move γ_1 for Player 1 that ensures the following: no matter how Player 2 plays, Player 1 knows, given the next observation, in which set of Y the next state of the game is.

This operator is used similarly as in the games of perfect information of previous section and so we have the following theorem.

Theorem 4 ([CDHR06, WDR06]) *Given a two-player game structure with imperfect information $(Q, q_0, \gamma_1, \gamma_2, \delta, \mathcal{O})$, a set of observations $\text{Good}^o \subseteq \mathcal{O}$, Player 1 has an observation based strategies to ensure the safety objective Good^o , iff $\{\{q_0\}\} \preceq \vee \{Y \mid Y = \{\text{Good}^o\} \wedge \text{Cpre}_1^o(Y)\}$*

References

- [AB06] Eugene Asarin and Patricia Bouyer, editors. *Formal Modeling and Analysis of Timed Systems, 4th International Conference, FORMATS 2006, Paris, France, September 25-27, 2006, Proceedings*, volume 4202 of *Lecture Notes in Computer Science*. Springer, 2006.
- [AHK02] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.
- [CDHR06] Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, and Jean-François Raskin. Algorithms for omega-regular games with imperfect information. In *CSL06*, volume 4207 of *Lecture Notes in Computer Science*, pages 287–302. Springer, 2006.
- [dAH00] Luca de Alfaro and Thomas A. Henzinger. Concurrent omega-regular games. In *LICS*, pages 141–154, 2000.
- [dAHM01] Luca de Alfaro, Thomas A. Henzinger, and Rupak Majumdar. From verification to control: Dynamic programs for omega-regular objectives. In *LICS*, pages 279–290, 2001.
- [HT06] João P. Hespanha and Ashish Tiwari, editors. *Hybrid Systems: Computation and Control, 9th International Workshop, HSCC 2006, Santa Barbara, CA, USA, March 29-31, 2006, Proceedings*, volume 3927 of *Lecture Notes in Computer Science*. Springer, 2006.
- [WDR06] Martin De Wulf, Laurent Doyen, and Jean-François Raskin. A lattice theory for solving games of imperfect information. In *HSCC 2006*, volume 3927 of *Lecture Notes in Computer Science*, pages 153–168. Springer, 2006.