

Computing Weakest Strategies for Safety Games of Imperfect Information

Wouter Kuijper and Jaco van de Pol

University of Twente **
Formal Methods and Tools
Dept. of EEMCS
{W.Kuijper, J.C.vandePol}@ewi.utwente.nl

Abstract. CEDAR (Counter Example Driven Antichain Refinement) is a new symbolic algorithm for computing weakest strategies for safety games of imperfect information. The algorithm computes a fixed point over the lattice of *contravariant antichains*. Here contravariant antichains are antichains over *pairs* consisting of an information set and an allow set representing the associated move. We demonstrate how the richer structure of contravariant antichains for representing antitone functions, as opposed to standard antichains for representing sets of downward closed sets, allows CEDAR to apply a significantly less complex controllable predecessor step than previous algorithms.

1 Introduction

Many problems related to synthesis and verification of systems reduce naturally to solving games [13, 8]. In particular, games of imperfect information form a class of games where the players have only partial access to the current state, which leads to the players having imperfect information about the game’s exact location. In our field this class of games is important since the concept of partial observability seems to arise quite naturally in several applications.

For example, the Controller Synthesis problem requires a control strategy for a plant to be automatically synthesized [9, 10]. Here, in general, not everything about the plant will be directly observable for the controller. On a more fine grained level of control, the problem of Motion Planning with Uncertainty naturally exhibits aspects of imperfect information. Finally, for a general problem like Interface Compatibility Checking [5] on componentized, object-oriented programs, we may consider the private fields and methods of an object leading to internal, non-observable behaviour.

When we restrict ourselves to safety objectives but maintain partial observability we are dealing with safety games of imperfect information. For this class, if a game is solvable, there always exists a *weakest* solution which subsumes all other solutions to the same game.

** This work was partly funded by NWO project 600.065.120.24N20

Complexity. It is known that partial observability bumps the complexity of two player games with perfect observation to a higher class [11]: the exponential complexity for solving games of imperfect information derives from an inherent subset construction needed to analyze the information sets of the player.

Recently, however, progress has been made in the form of symbolic algorithms that are able to analyze nondeterministic automata while avoiding the explicit subset construction for determinization [14, 15, 6]. One such new class of algorithms works by computing fixed points over antichains which are sets of pairwise incomparable sets of states (information sets). Although the inherent complexity of the problem still remains exponential, on instances the authors report significant efficiency gains [2].

Contribution. In our approach we limit the scope to safety objectives which allows for a symbolic algorithm that can compute weakest strategies. This is complementary to the approach of [4]. Our approach is useful for cases where (1) everything that one wants to synthesize is expressible as a safety property (e.g.: hard timeliness constraints for instance are expressible as a safety property) and/or (2) the result must be reusable and amenable to further analysis/composition/optimization. It is especially in case (2) where weakest strategies really shine.

Since a weakest strategy for a given game subsumes all possible safe strategies it is useful as a *safety monitor* (i.e: for supervising software or users that cannot be completely guaranteed safe). Computing the weakest strategy may also form part of a preprocessing step for generating a safe input graph to a second synthesis procedure that can optimize some performance measure that is not expressible as a safety property. Finally, weakest safety strategies are useful in a compositional setting where the behaviour of the context is not known beforehand so that a most general solution is necessary in order not to exclude possible safe compositions with a concrete context.

Our main contribution is a new algorithm named CEDAR (Counter Example Driven Antichain Refinement). In a nutshell, the algorithm computes a fixed point over an enriched form of antichains which we call *contravariant antichains*. Contravariant Antichains enjoy most of the properties of normal antichains, but they can represent knowledge based strategies, which are antitone functions from information sets to allow sets, rather than just sets of downward closed information sets. This additional structure allows us to symbolically compute, not just the set of winning initial information sets, but the entire weakest knowledge based strategy.

As a second contribution our approach permits to significantly simplify the controllable predecessor step. In contrast to [4] we only treat a single counterexample observation to the observation-closedness condition for the contravariant antichain, as opposed to treating all counterexample observations at every iteration.

Related Work. A result that contrasts with the symbolic approach is [3]. Here games are solved by searching the knowledge based subset construction in a forward direction (starting from the initial information set). The winning strategy is constructed while traversing this graph using an efficient on-the-fly fixed point algorithm due to [7]. This means that losing states are pruned out and back propagated at an early stage but it does

not constitute a fully symbolic algorithm since the algorithm still explicitly constructs (a subgraph of the) knowledge based subset construction.

The algorithm presented in [4] works for omega regular winning conditions, which from one point of view makes it more general than CEDAR, which works only for safety objectives. However this generality also comes at a price since for omega regular objectives there is in general no *weakest* strategy [1]. Indeed the algorithm computes the set of winning information sets, i.e.: the weakest information sets from which there still exists a winning strategy.

Recently the same authors show that the antichain representation of the largest winning regions for a given parity game does not allow to recover the winning strategy directly [2]. The authors present an algorithm that can construct a winning strategy using antichains as the underlying datastructure for representing sets of downward closed state sets. Clearly, since they are dealing with parity games, the algorithm will construct a strategy that is not necessarily the weakest. This approach can be seen as complementary to ours. Our approach is limited to safety games, but computes the strategy directly in the form of a contravariant antichain, and ensures that the resulting strategy is the weakest.

Structure of the paper. The paper is structured as follows. In Section 2 we define and discuss imperfect information safety games, strategies, and weakest strategies. In Section 3 we introduce contravariant antichains which is the new datastructure underlying CEDAR. In Section 4 we present the CEDAR algorithm. And finally in Section 5 we give preliminary experimental results and concluding remarks.

2 Safety Games of Imperfect Information

In this section we introduce formally the notion of a *safety game of imperfect information*, and we define the *weakest, antitone knowledge based strategy* for a given game.

Definition 1 (Safety Games) A *safety game of imperfect information* G is a tuple

$$G = (L, C^{\text{out}}, C^{\text{in}}, \alpha, \beta, \delta, i^{\text{init}})$$

consisting of a finite set of *game locations* L , a finite set of *control outputs* C^{out} , a finite set of *control inputs* C^{in} , an *output labeling* $\alpha : L \rightarrow C^{\text{out}}$, an *input labeling* $\beta : L \rightarrow C^{\text{in}}$, a *game board* $\delta \subseteq L \times L$, and a set of *initial locations* $i^{\text{init}} \subseteq L$ (also called the *initial information set*). We define $O = C^{\text{out}} \times C^{\text{in}}$ as the set of *observations*, an observation $o \in O$ is written as $o = c^{\text{out}}/c^{\text{in}}$. As a convenience we define labeling $\gamma : L \rightarrow O$ such that $\gamma(\ell) = \alpha(\ell)/\beta(\ell)$. We define $A = 2^{C^{\text{out}}}$ as the set of *allow sets*. Let $\alpha^{-1}(a) = \{\ell \in L \mid \exists c^{\text{out}} \in a. \alpha(\ell) = c^{\text{out}}\}$, and $\gamma^{-1}(o) = \{\ell \in L \mid \gamma(\ell) = o\}$; since it is always clear from the context where a set of locations is required, throughout the paper we will leave the conversions $\alpha^{-1}(\cdot)$ and $\gamma^{-1}(\cdot)$ implicit. \triangleleft

A safety game of imperfect information should be interpreted as a game between two players: *the safety player* and *the reachability player*. The objective for the safety

player is to keep the game running forever. The objective for the reachability player is to reach a deadlock state, i.e.: a location ℓ in which it holds $\delta(\ell) = \emptyset$.

Since we aim for a framework where strategies are ordered with respect to permissiveness we introduce moves for the safety player as *allow sets*. In this way we can have a subsumption relation on the moves for the safety player. The moves for the reachability player are then the *concrete successor locations* that are allowed by the game board *and* by the allow set chosen by the safety player. For example, if we are in game location $\ell \in L$ and the safety player chooses move $a \in A$, the reachability player must choose a successor location from the *forcing set* which is defined as $\delta(\ell) \cap a$. It is up to the safety player to ensure that her forcing set never becomes empty. Below we illustrate the definition with a concrete example of a safety game.

Example 1 (Pennymatching) We introduce a simple game of *penny-matching*. In this game, at each round, both players choose a side to a penny. If the safety player forfeits her choice by playing $a = \{h, t\}$ (heads *or* tails) the reachability player will choose for her. This may seem counterintuitive on this simple example, however note that, from a control perspective, this is a reasonable model: when the safety player permits two possible, distinct control outputs and the game-board does not resolve this choice either, she automatically yields this forcing power to her opponent.

The rules of the game are now as follows: if both players play heads the game is over and it is a win for the reachability player, in all other cases the game simply continues. To make the game slightly more interesting we stipulate that the reachability player cannot surprise the safety player by playing heads twice in a row.

Finally, in order to investigate the effect of imperfect information, we introduce two variants of the game: *open* pennymatching where the safety player can observe the coin of the reachability player, and, the harder variant, *blind* pennymatching where the safety player has no information about what side the reachability player chooses at each turn. According to definition 1 we may model these games as follows:

$L_{\text{penny}} = \{h, t\} \times \{h, t\}$	$C_{\text{open}}^{\text{in}} = \{h, t\}$	$C_{\text{blind}}^{\text{in}} = \{\mathbf{x}\}$	$C_{\text{penny}}^{\text{out}} = \{h, t\}$
$i_{\text{penny}}^{\text{init}} = \{ht\}$	$\beta_{\text{open}}(sr) = r$	$\beta_{\text{blind}}(sr) = \mathbf{x}$	$\alpha_{\text{penny}}(sr) = s$
$\delta_{\text{penny}} = \{(sr, s'r') \in L \times L \mid \neg(s = r = h) \wedge (r = h \rightarrow r' \neq h)\}$			

Note that we consistently shorten a location $(s, r) \in L_{\text{penny}}$ as a juxtaposition sr . In Figure 1 we show a fragment of the unraveling of this game into a *game dag*. The intermediate forcing sets (where the reachability player will choose his move) are shown as dotted boxes. Note that the move $a = \{h, t\}$ (the weakest move for the safety player) played from the initial game state transitively leads to all four possible game locations including the deadlock at ‘hh’. Further note that, played from the location ‘th’ the same move transitively leads to ‘ht’ or ‘tt’ which are both still safe.

The dashed lines connecting two nodes of the game dag indicate that for the *blind* version of the game these states in the unraveling of the game are *indistinguishable* for the safety player. Note that for the *open* version of the game it is immediately clear what would be the rational strategy for the safety player, the safety player just has to avoid the deadlock state marked with \times hence she has to play $a = \{t\}$ all the time until she observes ‘t/h’ after which she may relax her move to $a = \{h, t\}$. For the *blind* version

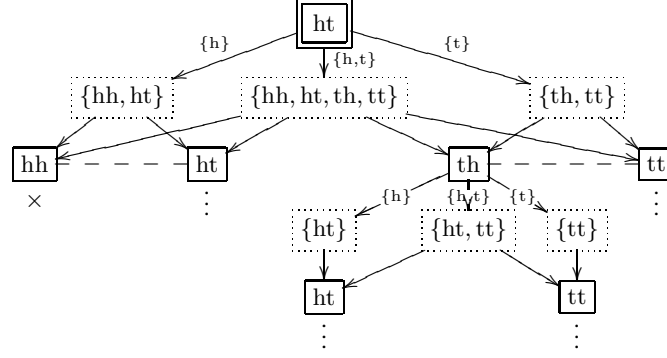


Fig. 1. Game DAG for the pennymatching game of Example 1.

of the game this is not so straightforward since her observation t/x cannot distinguish between the successor locations th and tt , and for that reason she can never know for sure to be in location th . How this is analyzed formally is shown in Example 2. \triangleleft

Knowledge Based Subset Construction. So far we have not explicitly dealt with the fact that the safety player has only a limited number of observations at her disposal. The fact that the safety player can only make a limited observation of the current state is commonly referred to as *partial observability*. Partial observability leads to the safety player having only *imperfect information* about the exact location of the game. The impact of imperfect information in the analysis of games is huge due to the fact that the game graph δ is not *observation deterministic*. This means that distinct branchings in δ are not always distinguishable for the safety player. It is well known that this type of non-determinism can be resolved by applying a subset construction. We give the definitions below. Recall that, for a given location, $\gamma(\ell) = o$ denotes the observable information on ℓ , in this sense the set of observations O partitions L .

Definition 2 (Knowledge Based Subset Construction) For a given game, with I we denote the set of *information sets* defined as $I = 2^L$, and with Δ we denote the *knowledge based subset construction* which is defined as a graph over information sets $\Delta \subseteq I \times I$ as follows:

$$\Delta = \{(i, i') \in I \times I \mid \exists o \in O. i' = \delta(i) \cap o\}$$

Note that the image of δ on i is $\delta(i) = \bigcup_{\ell \in i} \delta(\ell)$, now $i' = \delta(i) \cap o$ represents the strongest knowledge the safety player has about the successor location upon observing o with knowledge i about the source location. \triangleleft

Example 2 (Knowledge Based Subset Construction) Figure 2 shows a fragment of the knowledge based subset construction for the *blind* version of the pennymatching game from Example 1. We do not normally draw the empty information set, we do

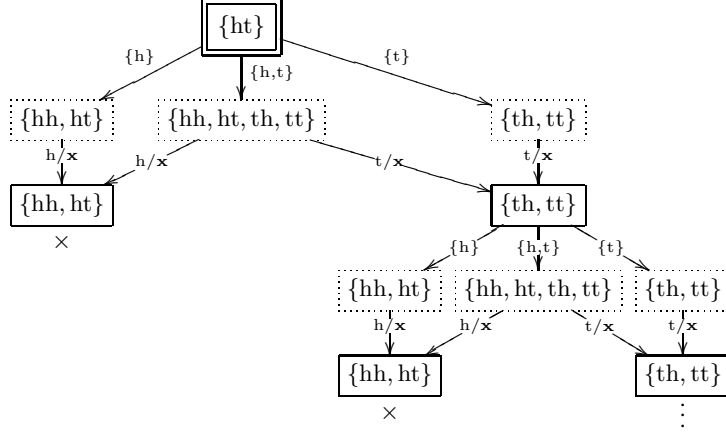


Fig. 2. Knowledge based subset construction for the blind pennymatching game.

include the intermediate forcing sets again for clarity, and now, in addition, we also show the observations that result from the moves for the reachability player.

From this graph it is clear what is the rational strategy for the blind pennymatching game: always play $a = \{t\}$ to avoid information sets that include a deadlock state. \triangleleft

Knowledge Based Strategies. We are now in a position to introduce the concept of a strategy for the safety player. This definition also determines the winning condition: the safety player wins the game iff she has a strategy to force an infinite play.

Definition 3 For a given game, a *knowledge based strategy* is a function $f : I \rightarrow A$. With F we denote the set of all knowledge based strategies. For a given strategy $f \in F$ and information set $i_0 \in I$, with $\text{outcome}(G, f, i_0)$ we denote the *outcome of f on G starting from i_0* as a set of non-empty traces of game locations annotated with information states: $\text{outcome}(G, f, i_0) \subseteq (L \times I)^+$. This is defined as follows:

$$\begin{aligned} \text{outcome}(G, f, i_0) = \{ & \ell_0 i_0 \dots \ell_n i_n \mid \forall j \leq n. \ell_j \in i_j, \text{ and } \forall j < n. \\ & (\ell_j, \ell_{j+1}) \in \delta \text{ and } (i_j, i_{j+1}) \in \Delta \text{ and } \alpha(\ell_{j+1}) \in f(i_j) \} \end{aligned}$$

These are all possible finite (partial) plays that may arise when our safety player is playing according to knowledge based strategy f . An outcome is *safe* iff no play ends in a deadlock (every finite play has a proper extension). We say that a strategy f is *safe* for G iff for all $i \in I$ either $\text{outcome}(G, f, i)$ is safe, or $f(i) = \emptyset$. A strategy is *winning* iff it is safe and $f(i^{\text{init}}) \neq \emptyset$. A game is *solvable* iff it permits a winning strategy. \triangleleft

An Inductive Definition of Safety. For the exposition of CEDAR we need to give an equivalent, inductive characterization of safety in terms of the following two elementary properties of knowledge based strategies. The first property is *obstinacy*, intuitively a

strategy is obstinate if it blocks completely on information sets for which an empty forcing set is possible, or, equivalently, it returns a non-empty allow set only if each of the states in the information set has at least one valid successor in the underlying game board intersected with the allow set. The second property is *observation-closedness*, intuitively a strategy is observation-closed if it can guarantee that non-blocking states will, for every possible observation, lead to non-blocking successor states.

One may think of these two properties as an inductive definition of safety where obstinacy forms the base case, and observation-closedness forms the inductive case.

Definition 4 (Inductive Safety) For a given game, a knowledge based strategy $f \in F$ is *obstinate* iff for all $i \in I$ such that there exists $\ell \in i$ for which $\delta(\ell) \cap f(i) = \emptyset$ it holds $f(i) = \emptyset$. A knowledge based strategy $f \in F$ is *observation-closed* iff for all $i \in I$ and $o \in O$ such that $\delta(i) \cap f(i) \cap o \neq \emptyset$ it holds that $f(\delta(i) \cap o) \neq \emptyset$. \triangleleft

The following lemma establishes that *obstinacy* and *observation-closedness* are sufficient conditions to characterize safety for knowledge based strategies.

Lemma 1 (Inductive Safety) For a given game, a strategy is safe iff it is both obstinate and observation-closed. \triangleleft

Weakest Strategies. In the previous sections we have consistently defined a solution to a safety game as *any* winning strategy. In this section we sharpen this to *the weakest*, or *most permissive* winning strategy. Intuitively, a winning strategy is the most permissive winning strategy if for all plays the strategy always yields the largest possible allow set that is sufficient for keeping the future play safe. Formally, this means we introduce an ordering on F with respect to which we may select the greatest element in the subset of safe strategies.

Definition 5 For a given game, we define a weak partial order \supseteq on F such that $f' \supseteq f$ iff for all $i \in I$ it holds $f'(i) \supseteq f(i)$. We say f' is *weaker* or *more permissive* than f . A strategy $f \in F$ is *antitone* iff for all $i, i' \in I$ it holds: $i \subseteq i'$ implies $f(i) \supseteq f(i')$. \triangleleft

We first show that for obtaining weakest, safe strategies, we can restrict our attention to antitone strategies.

Lemma 2 For a given game, for any safe strategy f there exists a weakest, safe, antitone strategy f' such that $f' \supseteq f$. \triangleleft

Proof Given a strategy f that is obstinate and observation-closed, we can define $g(i) := \bigcup \{f(i') \mid i \subseteq i'\}$. It is straightforward to show that $g \supseteq f$, and g is antitone, obstinate, and observation-closed.

Given any two antitone, obstinate and observation-closed strategies f_1 and f_2 , it can be checked that their join, defined as $(f_1 \sqcup f_2)(i) := f_1(i) \cup f_2(i)$ is also antitone, obstinate, and observation-closed. Hence, as the lattice of antitone safe strategies is finite, it is a complete lattice, and the weakest safe antitone $f' \supseteq g$ exists. \square

We can summarize this discussion by the following definition and theorem:

Definition 6 With f_G we denote the weakest, safe, antitone strategy on game G . \triangleleft

Theorem 3 For any game G it holds that G is solvable iff $f_G(i^{\text{init}}) \neq \emptyset$. \triangleleft

3 A Datastructure for Representing Antitone Functions

In this section we develop an efficient, symbolic representation for antitone functions as *contravariant antichains* which are antichains over domain/codomain pairs. First we give the general definition of a contravariant antichain, next we instantiate this definition and use it as the main datastructure underlying CEDAR. As it turns out, contravariant antichains are suitable for representing both knowledge based strategies as well as the set of open counterexample observations.

Definition 7 (Contravariant Antichains) Let (S, \subseteq^s) be some finite, partially ordered domain that forms a complete lattice, and (T, \subseteq^t) some finite, partially ordered codomain that forms a complete lattice. A *contravariant relation* is a set $h \subseteq S \times T$ that represents a function $\llbracket h \rrbracket : S \rightarrow T$ such that $\llbracket h \rrbracket(s) = \bigcup \{t' \mid \langle s', t' \rangle \in h, s \subseteq^s s'\}$, i.e.: an element s from the domain S is *implicitly* mapped to an element t of the codomain T that is the join of all t' to which weaker s' than s are *explicitly* related in h . We will frequently abuse notation and write simply $h(s)$ instead of $\llbracket h \rrbracket(s)$. We define the weak partial order $\subseteq^{(s,t)}$ on $S \times T$ as the product order: $\langle s, t \rangle \subseteq^{(s,t)} \langle s', t' \rangle$ iff $s \subseteq^s s'$ and $t \subseteq^t t'$. The corresponding strict partial order is denoted by $\subset^{(s,t)}$.

A *contravariant antichain* is a contravariant relation consisting of pairwise $\subset^{(s,t)}$ -incomparable domain/codomain pairs. With $\mathcal{C}[S, T]$ we denote the set of contravariant antichains from S to T . \triangleleft

Below we give an example of the use of contravariant antichains for representing knowledge based strategies.

Example 3 (Strategies as Contravariant Antichains) Assuming the definition in example 1, the following contravariant antichain $h_{\text{penny}} \in \mathcal{C}[I, A]$ represents a knowledge based strategy for the pennymatching game:

$$h_{\text{penny}} = \{\langle \{ht, tt, th\}, \{t\} \rangle, \langle \{th\}, \{h, t\} \rangle\}$$

Note that $\llbracket h_{\text{penny}} \rrbracket$ is a winning strategy for the (blind) pennymatching game. For this specific instantiation of Definition 7 we will refer to the domain/codomain pairs as *info/allow* pairs.

It is clear that contravariant antichains with their semantics in the domain of antitone functions form an adequate representation of knowledge based strategies. They are, however, not *canonical*. To see this note that the following contravariant antichain k_{penny} is equivalent to h_{penny} in the sense that they represent the same strategy:

$$k_{\text{penny}} = \{\langle \{ht, tt, th\}, \{t\} \rangle, \langle \{th\}, \{h\} \rangle\}$$

Note that $\llbracket k_{\text{penny}} \rrbracket(\{th\}) = \{h\} \cup \{t\} = \{h, t\}$. \triangleleft

Apparently there is still structure in a contravariant antichain. To characterize it, we lift \subseteq^s and \subseteq^t to *preorders* on $S \times T$, by defining $\langle s, t \rangle \subseteq^{(s,\cdot)} \langle s', t' \rangle$ iff $s \subseteq^s s'$, and similar for $\subseteq^{(\cdot,t)}$. Note that for the corresponding strict partial orders, we have: $\subset^{(s,t)} = (\subset^{(s,\cdot)} \cap \subset^{(\cdot,t)}) \cup (\subseteq^{(s,\cdot)} \cap \subset^{(\cdot,t)})$.

We now propose two possible canonical classes of contravariant antichains called *saturated contravariant antichains* and *sparse contravariant antichains*, respectively. A contravariant antichain is called *saturated* if it contains all $\subseteq^{(s,t)}$ -maximal domain/codomain pairs in the graph of the antitone function it represents. The strategy h_{penny} from example 3 is saturated. A contravariant antichain is *sparse* if it contains only $\subseteq^{(s,\cdot)}$ -principal pairs, which are all pairs for which the target is disjoint from the joined targets of all domain/codomain pairs that have a weaker source element. The strategy k_{penny} from example 3 is sparse. Both on saturated and sparse instances, $\subseteq^{(s,\cdot)}$ is anti-symmetric.

A contravariant antichain in its sparse normal form is generally smaller because it only contains pairs that have “added value”. However, in principle, it carries the same information as a contravariant antichain in its saturated normal form. In Section 4 we show how both normal forms are useful in practice.

Definition 8 (Sparse Contravariant Antichains) Let (S, \subseteq^s) , (T, \subseteq^t) be complete, finite lattices. For a given contravariant relation $h \subseteq S \times T$ and $s \in S$ with $h \uparrow s$ we denote h above s , defined as $h \uparrow s = \{\langle s', t' \rangle \in h \mid s \subseteq^s s'\}$. With $\mathbb{S}(h)$ we denote the *source set* of h defined as $\mathbb{S}(h) = \{s \in S \mid \exists t. \langle s, t \rangle \in h\}$. With $\llbracket h \rrbracket$ we denote the *sparse normal form* of h . This is defined as follows:

$$\llbracket h \rrbracket = \{\langle s, t \rangle \mid s \in \mathbb{S}(h) \text{ and } t = \llbracket h \rrbracket(s) \setminus \llbracket h \uparrow s \rrbracket(s) \text{ and } t \neq \emptyset\}$$

We say h is *sparse* iff $h = \llbracket h \rrbracket$. With $\llbracket \mathcal{C} \rrbracket[S, T]$ we denote the set of all sparse contravariant antichains from S to T . \triangleleft

Definition 9 (Saturated Contravariant Antichains) Given any contravariant relation $h \subseteq S \times T$, with $\llbracket h \rrbracket$ we denote the *restriction of h to $\subseteq^{(s,t)}$ -maximal elements*. This is defined as follows:

$$\llbracket h \rrbracket = \{\langle s, t \rangle \in h \mid t \neq \emptyset \text{ and } \nexists \langle s', t' \rangle \in h. \langle s, t \rangle \subset^{(s,t)} \langle s', t' \rangle\}$$

We define the *contravariant closure* as follows:

$$\llbracket \llbracket h \rrbracket \rrbracket = \llbracket \{\langle s, t \rangle \mid \exists \langle s_1, t_1 \rangle, \dots, \langle s_m, t_m \rangle \in h. s = \bigcap_{1 \leq j \leq m} s_j \text{ and } t = \bigcup_{1 \leq j \leq m} t_j\} \rrbracket$$

i.e.: for any non-empty subset of domain/codomain pairs we take the meet of the source elements and the join of the target elements. We say h is *saturated* iff $h = \llbracket \llbracket h \rrbracket \rrbracket$, with $\llbracket \mathcal{C} \rrbracket[S, T]$ we denote the set of all saturated contravariant antichains from S to T . For $h, k \in \llbracket \mathcal{C} \rrbracket[I, A]$ we let $h \sqcup k$ be the *join* of h and k , defined as $h \sqcup k = \llbracket h \cup k \rrbracket$. \triangleleft

4 An Algorithm for Computing Weakest Strategies

Algorithm 1 computes the weakest, safe knowledge based strategy for a given safety game of imperfect information. The algorithm works by approximating from above an obstinate, observation-closed fixed point in the lattice of saturated contravariant antichains. We recall our characterization of safety in terms of obstinacy and observation-closedness in Definition 4. The algorithm is based on the fact that we can maintain *obstinacy* as an invariant by never allowing any source location with empty forcing sets

into the strategy. *Observation-closedness* then remains as the fixed point condition that the algorithm needs to work towards. The idea is to approach the fixed point by treating, at each iteration, a counterexample against observation-closedness. A counterexample against observation-closedness consists of an information set $i \in I$ and an observation $o \in O$, such that $\delta(i) \cap f(i) \cap o \neq \emptyset$ and $f(\delta(i) \cap o) = \emptyset$, i.e., o can actually be observed, but the strategy blocks on the resulting information set.

In order to avoid an explicit iteration over all possible observations, CEDAR computes, for a given $i \in I$, the set of counterexample observations *symbolically*. To show how this is done we now give an alternative characterization of the set of counterexample observations as the set of *unexplained observations*.

Intuitively, an observation $o \in O$ from an information set $i \in I$ is *explained* by another information set $i'' \in I$, if the successor information set $i' = \delta(i) \cap o$ is a subset of i'' and $f(i') \neq \emptyset$. Note that, since f is antitone, it follows that $f(i') \neq \emptyset$ hence (i, o) is *not* a counterexample to observation-closedness. The set of observations that are *not explained* by any i'' can now be computed symbolically as

$$O^\dagger = \bigcap_{i'' \in I, f(i'') \neq \emptyset} \gamma((\delta(i) \cap f(i)) \setminus i'') \quad (1)$$

That is: O^\dagger contains all observations for which the successor information set from i is not completely inside any suitable i'' . In Definition 10, we see how the contravariant antichain representation of strategies simplifies the intersection in Equation 1 further.

Prerequisite Functions. Before we discuss CEDAR in more detail we first define the three helper functions in terms of which the algorithm is expressed.

Definition 10 (Unexplained Observations) For $h \in \llbracket \mathcal{C} \rrbracket [I, A]$ we let \widehat{h} be the antichain of maximal information sets in h , defined as: $\widehat{h} = \{i'' \in S(h) \mid \nexists i' \in S(h). i'' \subset i'\}$. We let $\text{Uobs}(h) \in \llbracket \mathcal{C} \rrbracket [I, 2^O]$ be the set of *unexplained observations*, defined as follows:

$$\text{Uobs}(h) = \llbracket \{ \langle i, O^\dagger \rangle \mid i \in S(h) \text{ and } O^\dagger = \bigcap_{i'' \in \widehat{h}} \gamma((\delta(i) \cap h(i)) \setminus i'') \} \rrbracket \quad (2)$$

Note the restriction to maximal information sets in Equation 2. This is valid since an observation that is not explained by any of the maximal information sets will certainly not be explained by any of the weaker information sets. Below we give an example of how defining Equation 2 is used to compute the set of counterexample observations.

Example 4 (Unexplained Observations for Pennymatching) We let $h \in \llbracket \mathcal{C} \rrbracket [I, A]$ be $h = \{\langle \{ht, tt, th\}, \{h, t\} \rangle\}$, this is the weakest obstinate strategy for the penny-matching game. So let $i = \{ht, tt, th\}$ and $a = \{h, t\}$. To compute $\text{Uobs}(h)$ we first compute the forcing set $\delta(i) \cap a = \{hh, ht, tt, th\}$. We then compute the set of unexplained observations, we have only one maximal information set $i'' = \{ht, tt, th\} \in \widehat{h}$, we subtract i'' from the forcing set and obtain $\{hh\}$. For this set we compute the set of corresponding observations $\gamma(\{hh\}) = \{h/x\}$. Since there is only one maximal information set, in this case, the intersection is trivially done: for i we get simply $O^\dagger = \{h/x\}$, and hence $\text{Uobs}(h) = \{\langle \{ht, tt, th\}, \{h/x\} \rangle\}$. \triangleleft

After explaining how to detect counter examples as unexplained observations, we now explain how to treat such a counterexample (i, o) . First, let $h \Downarrow i$ be the *substrategy of h on i* , defined as $h \Downarrow i = \{\langle i', a' \rangle \in h \mid i' \subseteq i\}$. This represents the behaviour of the strategy $\llbracket h \rrbracket$ on all the information sets that are stronger-than-or-equal-to i . Now, if i has an unexplained observation $c^{\text{out}}/c^{\text{in}} \in \llbracket \text{Uobs}(h) \rrbracket(i)$, since all *stronger* $i' \subseteq i$ have a *weaker* allow set $h(i') \supseteq h(i)$ these stronger i' will also allow c^{out} . Hence, to effectively “treat” the unexplained observation CEDAR will replace the entire affected substrategy by the most permissive substrategy that is observation-closed on $o = c^{\text{out}}/c^{\text{in}}$. This most permissive substrategy will actually be the join of two substrategies, which are based on the restricted successor, and the controllable predecessor.

The controllable predecessor substrategy, illustrated in Figure 3 (a), contains the weakest, obstinate info/allow pairs that explain the observation by strengthening the information set to the weakest controllable region that forces the successor information set within one of the existing maximal information sets $i'' \in \hat{h}$, i.e.: this new substrategy solves the problem by requiring *more knowledge*.

Definition 11 (Controllable Predecessor) For some strategy $h \in \llbracket \mathcal{C} \rrbracket[I, A]$, a set of maximal information sets $q \subseteq I$, and some observation $o = c^{\text{out}}/c^{\text{in}} \in O$ we let $\text{Cpre}(h, q, o) \in \llbracket \mathcal{C} \rrbracket[I, A]$ be the *controllable $c^{\text{out}}/c^{\text{in}}$ -predecessor strategy of h* , defined as follows:

$$\text{Cpre}(h, q, o) = \llbracket \{\langle i^c, a \rangle \mid \langle i, a \rangle \in h, i'' \in q, i^c = i \setminus \delta^{-1}((\delta(i) \cap o) \setminus i'')\} \rrbracket$$

The restricted successor substrategy, illustrated in Figure 3 (b), contains the weakest, obstinate info/allow pairs that avoid the unexplained observation by restricting the allow set and hence preventing the observation from arising at all, i.e.: this new substrategy solves the problem by becoming *less permissive*.

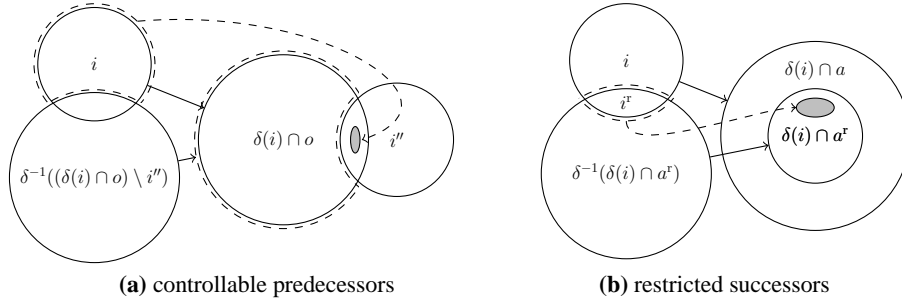


Fig. 3. Illustration of Definitions 11, 12: **(a)** for a given i, o and i'' we compute the weakest $i^c \subseteq i$ such that $\delta(i^c) \cap o \subseteq i''$, i.e.: i'' explains (i^c, o) . Here, i^c is the left dashed region, and the gray region denotes $\delta(i^c) \cap o$. **(b)** for a given i, a and $a^r \subset a$ we compute the weakest $i^r \subseteq i$ such that $\langle i^r, a^r \rangle$ is obstinate. The gray region denotes $\delta(i^r) \cap a^r$.

Definition 12 (Restricted Successor) We let $\text{Rsucc}(h, c^{\text{out}}) \in \llbracket \mathcal{C} \rrbracket [I, A]$ be the *restricted c^{out} -successor strategy of h* , defined as follows:

$$\text{Rsucc}(h, c^{\text{out}}) = \llbracket \{ \langle i^{\text{r}}, a^{\text{r}} \rangle \mid \langle i, a \rangle \in h, a^{\text{r}} = a \setminus \{c^{\text{out}}\}, i^{\text{r}} = i \cap \delta^{-1}(\delta(i) \cap a^{\text{r}}) \} \rrbracket$$

Description of the Algorithm. We have now all prerequisites to present Algorithm 1, and illustrate its working on the blind penny matching example.

In line 1 the contravariant antichain is initialized to be fully uninformed, except that the system is not initially in a deadlock state, and maximally permissive, i.e. it allows *all* control outputs. The while condition in line 2 states what is basically the negation of observation-closedness, in terms of Definition 10. In line 3 we select a counterexample information set and concrete observation from the (symbolic) set of its unexplained observations. In line 4 we compute the most conservative refinement needed to make the contravariant antichain observation-closed for the selected counterexample observation. The most permissive substrategy is computed based on Definitions 11 and 12.

Note that this refinement is *strict* since (1) in the restricted successor, the allow set a^{r} of each newly introduced pair is guaranteed to be a strict subset of a , and (2) in the controllable predecessor, the information set i^{c} for each newly introduced pair is guaranteed to be a strict subset of i . Further note that, for (1), using a *saturated* contravariant antichain for info/allow pairs makes sure that besides being *strict*, the restricted successor strategy is also the *most permissive*, and, for (2), using a *sparse* contravariant antichain for the counterexample pairs makes sure that the new i^{c} pairs are never fully absorbed by the existing pairs above i in the saturated strategy.

After one iteration of the while loop we have obtained the next contravariant antichain which is strictly below the previous one but always above-or-equal-to f_G in the strategy subsumption ordering \sqsubseteq . The algorithm terminates when there are no more counterexamples to observation-closedness. Since the other requirement on f_G , obstinacy, is an invariant of the algorithm it follows that, at termination, $\llbracket h \rrbracket = f_G$. In line 5 we do a final test to see if f_G is winning from the initial state, if so then the results are useful and h is returned. If we are not interested in any particular initial state we may set $i^{\text{init}} = \emptyset$ since, upon termination, it always holds: $h(\emptyset) = f_G(\emptyset) = C^{\text{out}}$.

Algorithm 1: CEDAR (Counter Example Driven Antichain Refinement)

Data: $G = (L, C^{\text{out}}, C^{\text{in}}, \alpha, \beta, \delta, i^{\text{init}})$ — a game.
Result: the contravariant antichain h such that $\llbracket h \rrbracket = f_G$, or \emptyset in case G is unsolvable.

- 1 $h \leftarrow \{ \{ \ell \in L \mid \delta(\ell) \neq \emptyset \}, C^{\text{out}} \}$
- 2 **while** $\text{Uobs}(h) \neq \emptyset$ **do**
- 3 **select some** $\langle i, O^\dagger \rangle \in \text{Uobs}(h)$ **and** $c^{\text{out}}/c^{\text{in}} \in O^\dagger$
- 4 $h \leftarrow (h \setminus h \Downarrow i) \sqcup (\text{Rsucc}(h \Downarrow i, c^{\text{out}}) \sqcup \text{Cpre}(h \Downarrow i, \widehat{h}, c^{\text{out}}/c^{\text{in}}))$
- 5 **if** $h(i^{\text{init}}) \neq \emptyset$ **then**
- 6 **return** h
- 7 **else**
- 8 **return** \emptyset

Example 5 (Solving pennymatching with CEDAR) The next table shows the successive values of the main program variables during a run of the algorithm on the blind pennymatching game of Example 1.

line 1; initialization:	$h \leftarrow \{\langle\{ht, th, tt\}, \{h, t\}\rangle\}$
line 2; observation closed?	$Uobs(h) = \{\langle\{ht, th, tt\}, \{h/x\}\rangle\}$
line 3; select counterexample:	$i \leftarrow \{ht, th, tt\}$ $c^{out}/c^{in} \leftarrow h/x$
line 4; refinement:	$Rsucc(h \Downarrow i, h) = \{\langle\{ht, th, tt\}, \{t\}\rangle\}$ $Cpre(h \Downarrow i, \hat{h}, h/x) = \{\langle\{th\}, \{h, t\}\rangle\}$ $h \leftarrow \{\langle\{ht, th, tt\}, \{t\}\rangle, \langle\{th\}, \{h, t\}\rangle\}$
line 2; observation closed?	$Uobs(h) = \emptyset$
line 5; strategy is winning?	$h(\{ht\}) = \{t\}$
line 6; yes, return h	$h = \{\langle\{ht, th, tt\}, \{t\}\rangle, \langle\{th\}, \{h, t\}\rangle\}$

As can be seen, on this simple example, the fixed point is reached after a single iteration. And the resulting strategy is indeed the *weakest* (and in this case *winning*) strategy. \triangleleft

5 Experiments and Conclusion

We made a prototype implementation of the algorithm using the BUDDY package [12] for BDD manipulations. For comparison, we also implemented an on-the-fly, forward fixed point evaluation over the knowledge based subset construction as described by [3]. We refer to that algorithm as OTFOE (On-The-Fly fOrward Exploration). The comparison is not completely fair, because OTFOE computes a partial weakest strategy only for *reachable* information sets, and, vice versa, cedar does not compute the reachable information sets. Although the algorithms compute slightly different results, still it is interesting to contrast the fully symbolic approach with the explicit forward exploration.

We evaluated both algorithms on two different architectures. The first architecture is a single *monolithic* game graph. The second architecture is a *composed* game graph generated by four randomly synchronizing components of which only the first one contains a deadlock and control in- and outputs. Both architectures are illustrated in Figure 4. The distinguishing difference of the compositional architecture, as opposed to the monolithic architecture, lies in the concurrency and locality exhibited by the former and not by the latter.

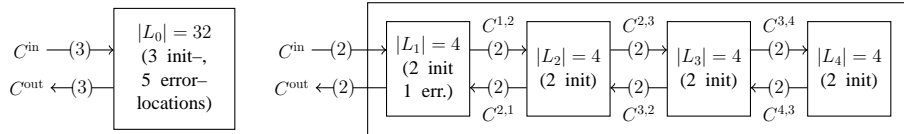


Fig. 4. The *monolithic* (left) and *compositional* (right) architectures. For the compositional architecture we make sure that component x is input enabled with respect to the internal synchronization labels in $C^{y,x}$. These special labels are projected away in the final game graph.

Table 1. Results for CEDAR and OTFOE on monolithic and compositional games. Under *ops.* we give the number of $\delta(\cdot)$ and $\delta^{-1}(\cdot)$ operations performed. Under *size* we give the *final (maximal)* size of the contravariant antichain for CEDAR and the *explored (safe)* information sets for OTFOE.

monolithic games					compositional games				
#	CEDAR		OTFOE		#	CEDAR		OTFOE	
	ops.	size	ops.	size		ops.	size	ops.	size
1	18	4(4)	327	105(69)	1	13	2(2)	23	8(0)
2	207	1(8)	11	4(0)	2	9	3(3)	116	43(30)
3	290	1(12)	265	69(0)	3	16	3(3)	200	68(24)
4	90	6(8)	340	103(62)	4	16	4(4)	17	7(4)
5	62	5(6)	303	98(60)	5	42	1(4)	260	81(0)
6	37	5(5)	229	78(42)	6	9	3(3)	219	80(60)
7	203	1(11)	11	4(0)	7	77	3(4)	27	10(0)
8	314	5(14)	20	7(0)	8	15	3(3)	13	5(3)
9	212	1(10)	183	44(0)	9	27	4(5)	39	15(9)

The game components are randomly generated with a fixed number of locations and labels (cf. Figure 4). Random δ relations are generated componentwise. We take care that each δ_x is input enabled so that the product δ is always deadlock free. We then introduce deadlocks on a random set of error locations. For the transition density, $r_\delta = |\delta|/|L|^2$, we maintain $r_\delta = 0.3$ for the monolithic architecture and $0.04 < r_\delta < 0.08$ for the compositional architecture. Around these values we observed the maximal number of safely reachable information sets on average. Note that this biases our experiments to dense, solveable game graphs.

We solved 9 random games for each architecture and measured the number of $\delta(\cdot)$ and $\delta^{-1}(\cdot)$ operations performed. These are principal operations for both CEDAR and OTFOE. In particular, OTFOE uses one $\delta(\cdot)$ for determining the forcing set of each newly generated information set, and one $\delta^{-1}(\cdot)$ to test obstinacy whenever an allow set changes (i.e. for new states and for backpropagating unsafe control outputs).

The results of the experiments are shown in Table 1 and Figure 5. We see that, on dense game graphs generated by many components, CEDAR performs better than

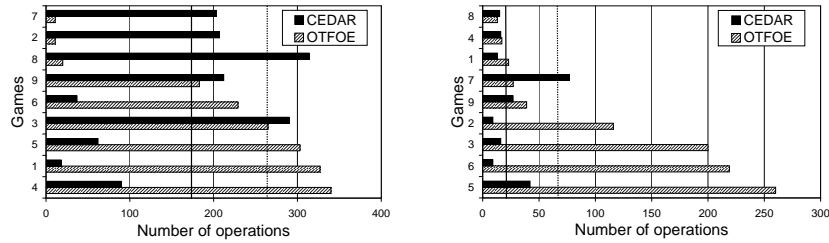


Fig. 5. Sorted charts of the data in Table 1; for *monolithic* (left) and *compositional* games (right). The dotted and solid lines show averages over 100 games for OTFOE and CEDAR, respectively.

OTFOE. We still see evidence of bad worst-case behaviour in the form of outliers like random compositional game number 7. For sparser game graphs we observed the worst performance of CEDAR around $r_\delta = 0.1$ for monolithic games ($\approx 3.3 \cdot 10^3$ operations on average), and around $r_\delta = 0.01$ for compositional games (≈ 46 ops.). Whether or not bad worst-case behaviour plays a significant role on real instances needs to be evaluated by testing the algorithm on real-world models.

As future work, we suggest to investigate the degrees of freedom allowed in CEDAR, for instance in selecting the next counter example. Dynamic programming techniques could speed up the implementation, by avoiding the complete recomputation of the next counter example. In order to preserve memory usage, one could also store the strategies in a *sparse* contravariant antichain, and recompute its saturated pairs in each iteration. Finally, one could compare the efficiency of CEDAR with the antichain method in [2]. However, a fair comparison is complicated because both algorithms compute essentially different objects. In a separate paper, we will show how the results can be used for compositional controller synthesis.

References

1. Julien Bernet, David Janin, and Igor Walukiewicz. Permissive strategies: from parity games to safety games. *Theoretical informatics and applications*, July 2008.
2. Dietmar Berwanger, Krishnendu Chatterjee, Laurent Doyen, Thomas Henzinger, and Sangram Raje. *Strategy Construction for Parity Games with Imperfect Information*, volume 5201/2008 of *LNCS*, pages 325–339. Springer, 2008.
3. Franck Cassez. Efficient on-the-fly algorithms for partially observable timed games. In *FORMATS 2007*, volume 4763 of *LNCS*, pages 5–24. Springer, 2007.
4. Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, and Jean-Francois Raskin. Algorithms for omega-regular games with imperfect information,. In *CSL 2006*, volume 4207 of *LNCS*, pages 287–302. Springer-Verlag, September 2006.
5. Luca de Alfaro and Thomas A. Henzinger. Interface automata. In *FSE*, pages 109–120. ACM Press, 2001.
6. Orna Kupferman, Nir Piterman, and Moshe Y. Vardi. Safrless compositional synthesis. In *CAV 2006*, volume 4144 of *LNCS*, pages 31–44. Springer, 2006.
7. Xinxin Liu, C. R Ramakrishnan, and Scott A Smolka. Fully local and efficient evaluation of alternating fixed points. In *TACAS 1998*, LNCS. Springer, 1998.
8. Ren Mazala. *Infinite Games*, pages 197–204. LNCS. Springer, 2002.
9. Amir Pnueli and Roni Rosner. On the synthesis of an asynchronous reactive module. In *ICALP 1989*, volume 372 of *LNCS*, pages 652–671. Springer, 1989.
10. P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM J. Control Optim.*, 25:206–230, 1987.
11. John H. Reif. The complexity of two-player games of incomplete information. *Journal of Computer and System Sciences*, 29:274–301, 1984.
12. Jørn Lind-Nielsen. Buddy: Binary decision diagrams. <http://sourceforge.net/projects/buddy>.
13. Jean Tirole and Drew Fudenberg. *Game Theory*. MIT Press, August 1991.
14. Martin De Wulf, Laurent Doyen, Thomas A. Henzinger, and Jean-Francois Raskin. Antichains: A new algorithm for checking universality of finite automata. In *CAV 2006*, volume 4144 of *LNCS*, pages 17–30. Springer, 2006.
15. Martin De Wulf, Laurent Doyen, and Jean-Francois Raskin. A lattice theory for solving games of imperfect information. In *Hybrid Systems: Computation and Control*, volume 3927 of *LNCS*, pages 153–168. Springer, 2006.