

Games with imperfect information

Dietmar Berwanger*

MPRI 2009/10

In our previous study on graph games, we assumed that players are perfectly informed about each move performed during the history of a play and, in particular, about the current position of the play. However, in real-world situations, it is often the case that a player is uncertain about the moves of other players, or even about the outcome of his own actions. For an example in computing, we may think of a process which should display a correct behaviour within a parallel system although it is unable to observe the private variables of other processes.

In this chapter, we study games with imperfect information where a player receives only partial information about the current state of the play.

1 One player against the environment

1.1 Model

The basic model involves just one player that interacts with the environment by performing one action in each stage and receiving in turn an observation about the current state of the play. To describe such a game, we fix a set A of *actions* and a set B of *observations* throughout the lecture, these sets are always finite.

A game graph (with imperfect information) is a structure $G = (V, \Delta, \beta)$ consisting of a set V of positions, a move relation $\Delta \subseteq V \times A \times V$ and an *observation* function $\beta : V \rightarrow B$. We may think of the move relation Δ as a set of edges labelled with actions $a \in A$. Again, we assume that every position has a outgoing move, i.e., $\Delta(v, a) = \{w : (v, a, w) \in \Delta\} \neq \emptyset$, for every $v \in V$. However, for a position v , there may be multiple outgoing moves with the same action label $a \in A$.

Intuitively, we can describe a play as a process of moving a token along the edges of G as follows: at the beginning the token is placed on the designated starting position v_0 . In each round, the player chooses an action a . Then, the environment moves the token from its current position v to one (of the possible many) a -successor $w \in \Delta(v, a)$. At this point, the player is not informed about the new position w of the play; instead, he receives the observation $\beta(w) \in B$.

*LSV, ENS Cachan – dwb@lsv.ens-cachan.fr – revision: February 7, 2010

For an observation $b \in V$, we call the set $V_b := \{v \in V : \beta(v) = b\}$ the observation set of b . Notice that the collection of observation sets forms a partition of V .

A play is thus described by an alternating sequence of positions and actions $\pi = v_0, a_0, v_1, a_1, v_2, \dots$; initial plays are play prefixes that end with a position. The information that a player receives during a play π (or an initial play) is the *observation trace* $\beta(\pi) := \beta(v_0), \beta(v_1), \beta(v_2), \dots$. His choices during the play can only rely on this trace. Accordingly, a strategy for the player of a game with imperfect information is defined as a mapping $s : (VA)^*V \rightarrow A$ that does not distinguish between initial plays with the same observation trace, i.e., for any two initial plays π, π' with $\beta(\pi) = \beta(\pi')$, we have $s(\pi) = s(\pi')$. We say that a play π follows s , if $a_\ell = s(v_0, a_0, \dots, v_{\ell-1})$ for all $\ell > 0$. The set of possible outcomes of a strategy is the set of all plays that follow it.

Notice that we can interpret a strategy as a function $s : B^* \rightarrow A$ mapping sequences of observations to actions. The definition on initial plays is chosen for convenience towards extending the model to more than one players.

A *winning condition* on G is a set $W \subseteq V^\omega$ of sequences of vertices. A strategy s is winning if, for every play $v_0, a_0, v_1, a_1, v_2, \dots$ that follows it, the sequence v_0, v_1, v_2, \dots belongs to W .

As in the case of games with perfect information, we consider game graphs equipped with a coloring function $\Omega : V \rightarrow C$, and specify winning conditions as a set $W \subseteq C^\omega$ standing for the condition

$$\{v_0, v_1, v_2, \dots : \Omega(v_0), \Omega(v_1), \Omega(v_2), \dots \in W\}.$$

Finally, a *game* $\mathcal{G} = (G, W)$ is described by a game graph and a winning condition.

- A *Reachability* condition is specified by a set $F \subseteq B$ of *target* observations that describes the winning condition

$$\{v_0, v_1, \dots : \beta(v_\ell) \in F \text{ for some } \ell \geq 0\}.$$

- A *Safety* condition is specified by a set $F \subseteq V$ of *safe* observations that describes the winning condition

$$\{v_0, v_1, \dots : \beta(v_\ell) \in F \text{ for all } \ell \geq 0\}.$$

A *parity* condition is specified by a *priority* function $\Omega : V \rightarrow \omega$ of finite range that describes the winning condition

$$\{v_0, v_1, v_2, \dots : \min \text{Inf}(\Omega(v_0), \Omega(v_1), \Omega(v_2), \dots) \text{ is even } \}.$$

One issue in games with imperfect information is whether the coloring that determines the winning condition is observable to the player. In this lecture we will generally consider game graphs with winning conditions that are *visible* in the following sense: for all $v, v' \in V$, if $\beta(v) = \beta(v')$ then $\Omega(v) = \Omega(v')$.

A game graph is of perfect information, if the set B of observations corresponds to the set V of positions and the observation function is the identity. In this case, the environment plays the role of the second player with the ability to choose the observation.

1.2 An example

Reachability game with two counter gadgets, counting modulo 3 and modulo 5, respectively. Two actions \circ – increments counters, and \bullet – leads to target when taken just before counter reset, otherwise leads to a sink and the player loses. Environment chooses one of the two counters at the beginning of the game; observation tells whether target is reached or not. Winning strategy: play \circ for (a multiple of) 15 rounds, then \bullet .

1.3 Solution procedure

The above example shows that winning strategies in a game with imperfect information may require memory.

1.3.1 Strategy automata

A strategy automaton is a Mealy machine $\mathcal{M} = (M, B, A, m_0, \mu, \nu)$ where

- M is a finite set of memory states with a designated initial state m_0 ,
- B is a set of observations,
- A is a set of actions,
- $\mu : M \times B \rightarrow S$ is a memory update function and
- $\nu : M \times B \rightarrow A$ is an action choice function.

We extend μ to sequences of observations by setting $\mu(m, \varepsilon) := m_0$ for the empty sequence ε , and $\mu(m, \tau \cdot b) = \mu(\mu(m, \tau), b)$ for any $\tau \in B^*$.

The automaton \mathcal{M} implements the strategy s for a game graph G with starting position v_0 if, for every initial play $v_0, a_0, v_1, \dots, v_\ell$,

$$s(v_0, a_0, v_1, a_0, \dots, v_\ell) = \nu(\mu(m_0, \beta(v_0))\beta(v_1)\beta(v_2) \cdots \beta(v_\ell), \beta(v_\ell)).$$

We say that a game \mathcal{G} requires memory of size n if the minimal number of states of an automaton that implements a winning strategy for this game is n .

1.3.2 The knowledge-set construction

We present a solution procedure proposed in (J. Reif, The complexity of two-player games of incomplete information, *Journal of Computer and System Sciences* 29(1984), pp. 274-301). The idea of this procedure is to keep track of the set of positions at which the token could be. Intuitively, (an abstraction of)

the knowledge that a player has about the play in a game \mathcal{G} is stored as a set of positions $q \subseteq V$; whenever the player takes an action $a \in A$ and receives an observation $b \in B$, the knowledge is updated to:

$$\text{post}(q, a) = \{w \in V : \text{there exists } v \in q \text{ with } (v, a, w) \in \Delta\}.$$

This gives rise to a game with perfect information, the solution of which can be translated back to the original game with imperfect information.

To describe the algorithm more formally, let \mathcal{G} be an imperfect-information game with a visible winning parity condition. We refer to any set $\mathcal{P}(V) \setminus \emptyset$ as a *cell*. Let Q be the set of all *cells*.

We construct a *knowledge* game \mathcal{G}^K with the same set A of actions as the \mathcal{G} , but with Q as a set observations, as follows. The graph $G^K = (Q, \Delta^K, \beta^K)$ of \mathcal{G}^K consists of

- a set $Q = \mathcal{P}(V) \setminus \emptyset$ of positions corresponding to cells,
- a set of moves $(q, a, q') \in \Delta^K$ if, and only if, there exists an observation $b \in B$ such that

$$q' = \text{post}(q, a) \cap V_b;$$

- and an observation function β^K taken as the identity function over Q .

If v_0 is the initial position in G , the initial cell in G^K is $q_0 = \{v_0\}$.

Notice that \mathcal{G}^K is a game of perfect information. Moreover, every cell reachable in \mathcal{G}^K from $\{v_0\}$ is contained in an observation set V_b for some $b \in B$. Hence, we can extend the coloring $\Omega : V \rightarrow C$ associated to any visible winning condition by setting $\Omega(q)$ to be the value $\Omega(v)$ for any representative $v \in q$. Thus, any visible winning condition on \mathcal{G} , in particular parity, extends naturally to \mathcal{G}^K .

Theorem 1. *Let \mathcal{G} be a parity game with imperfect information and let \mathcal{G}^K be the game with perfect information obtained through the knowledge-set construction.*

- *The player has a winning strategy in \mathcal{G} starting from v_0 if, and only if, Player 0 has a winning strategy in \mathcal{G}^K starting from $\{v_0\}$, and*
- *Every memoryless winning strategy of Player 0 in \mathcal{G}^K can be effectively transformed into a winning strategy in \mathcal{G} implemented by an automaton with at most $2^{|V|}$ many memory states.*

Proof. Let us assume that Player 0 has a winning strategy in \mathcal{G}^K from $\{v_0\}$. As this is a parity game with perfect information, there also exists a memoryless winning strategy $f : Q \rightarrow Q$. We construct the memory automaton $\mathcal{M} = (Q, Q, A, \{v_0\}, \mu, \nu)$ with $\mu(q, b) = \text{post}(q, f(q)) \cap V_b$ and $\nu(q, b) = f(q)$. Now, we claim that the strategy s implemented by \mathcal{M} is winning in the imperfect-information game \mathcal{G} .

We argue that the plays that follow s in \mathcal{G} visit the same sequence of colors as the plays that follow f in \mathcal{G}^K .

More precisely, we show that for every play $\pi = v_0, a_0, v_1, \dots$ in \mathcal{G} that follows s , there exists a play $\pi' = q_0, a_0, q_1, \dots$ in \mathcal{G}^K that follows f and visits with the same sequence priorities as π , that is $\Omega(\pi) = \Omega(\pi')$. To construct π' we simply set $q_\ell = \text{post}(q_{\ell-1}, a_{\ell-1}) \cap V_\beta(v_\ell)$. As π winning in \mathcal{G} and π' visits the same sequence of priorities, it follows that s is a winning strategy. Clearly, the memory automaton \mathcal{M} that implements s has at most $2^{|V|}$ many states.

Conversely, assume that there exists no winning strategy for the game \mathcal{G} starting from v_0 . In other words, every strategy s in this game has a possible play outcome that is not winning. For this case, we show that Player 0 does not have a winning strategy in \mathcal{G}^K .

Towards this, let f be an arbitrary memoryless strategy on \mathcal{G}^K , and construct a strategy s for the game \mathcal{G} (with imperfect information) by setting for every initial play π in this game $s(\pi) = f(K(\pi))$ with $K(\pi)$ – the knowledge after playing π – defined inductively by $K(v_0) := \{v_0\}$ and

$$K(v_0, a_0, v_1, \dots, v_\ell) := \text{post}(K(v_0, a_0, v_1, \dots, v_{\ell-1}), a_{\ell-1}) \cap V_\beta(v_\ell), \text{ for all } \ell > 0.$$

By assumption, there exists play $\rho = v_0, a_0, v_1, a_1, \dots$ in \mathcal{G} that follows s and is not winning. Let t be a strategy for Player 1 in \mathcal{G}^K that chooses the cell $K(v_0, a_0, v_1, a_1, \dots, v_\ell)$ at every initial play with actions $a_0, a_1, \dots, a_{\ell-1}$. Then, the play that follows f and t in \mathcal{G}^K has the same sequence of observations as ρ , hence it is not winning. Consequently the strategy f cannot be a winning strategy in \mathcal{G}^K . Since f was chosen arbitrarily, this implies that Player 0 does not have a winning strategy in \mathcal{G}^K . This concludes the proof. \square

The above construction yields a procedure for solving games with imperfect information: given a parity game \mathcal{G} with imperfect information, construct the associated knowledge game \mathcal{G}^K with perfect information and solve it, i.e, establish whether Player 0 has a winning strategy and construct one if this is the case; finally transfer the solution to the initial game \mathcal{G} . The running time of this procedure is exponential with respect to the number of positions in \mathcal{G} . As the following result from Reif (1984) shows, the worst-case complexity cannot be improved.

Proposition 2. *The problem of deciding whether the player in a game with imperfect information has a winning strategy is EXPTIME hard.*

However, as we briefly discussed in the lecture, there are methods that use antichains for an efficient representation of computations with cells which turn out to perform well in practical cases.

2 Distributed games

In the following section, we discuss games with imperfect information that involve more than one player. Towards this, we extend the model of a one-player

game to one where actions are performed simultaneously by all players and each player has his own observation function. Most basic notions from the one-player case generalise in a straightforward way to the n -player setting. We detail them just for completeness.

2.1 Games with several players and simultaneous moves

We consider games played by n players, $0, 1, \dots, n-1$, against the environment. Beforehand, we fix a set A^i of actions available to Player i and a set B^i of observations of Player i . We denote by A the set of action profiles and by B the set of observation profiles.

A *game graph* is a structure $G = (V, \Delta, (\beta^i)_{i < n})$ with a set V of positions, a move relation $\Delta \subseteq V \times A \times V$ and an *observation function* $\beta^i : V \rightarrow B^i$ for each player $i < n$. As usual, we assume that each position in a game graph has at least one outgoing move for every action profile, i.e., $\Delta(v, a) \neq \emptyset$, for all $v \in V$ and all $a \in A$.

During a play, the n players interact with the environment in forming a path by moving a token along the edges of the game graph as follows. At the beginning, the token is at a designated initial position $v_0 \in V$ known to all participants. The game is played in rounds; in every round, each player i chooses an action $a^i \in A^i$. Then, according to the current position v of the token and the joint profile a of actions, the environment chooses a successor $v' \in \Delta(v, a)$ to which the token is moved. Here, each player i receives the observation $\beta^i(v')$ and the play continues with v' as the current position. Notice that the players are neither informed about the current position of the token, nor about the action chosen by other players.

Formally, a *play* starting from a designated position $v_0 \in V$ in G is an infinite sequence $\pi = v_0, a_0, v_1, a_1, \dots$ alternating between positions and action profiles with $(v_\ell, a, v_{\ell+1}) \in \Delta$, for all $\ell \geq 0$. An *initial play* is a prefix $v_0, a_0, \dots, a_{\ell-1}, v_\ell$ of a play. We extend the observation functions from positions to (initial) plays $\pi = v_0, a_0, v_1, a_1, \dots$ by setting $\beta^i(\pi) = \beta^i(v_0), \beta^i(v_1), \dots$.

A *strategy* for Player i is a partial function $s^i : (VA)^*V \rightarrow A^i$ that assigns an action to every initial play such that $s^i(\pi) = s^i(\pi')$ for any two plays π and π' that induce the same sequence of observations $\beta^i(\pi) = \beta^i(\pi')$. We denote the set of all strategies of Player i with S^i and the set of all strategy profiles by S . A play (or an initial play) $\pi = v_0, a_0, v_1, a_1, \dots$ follows the strategy $s^i \in S^i$, if $a_\ell^i = s^i(v_0, a_0, v_1, \dots, a_{\ell-1}, v_\ell)$ for every index $\ell > 0$. For a coalition $I \subseteq \{0, \dots, n-1\}$, the play π follows a profile s^I of strategies, if it follows all strategies s^i of the players $i \in I$. The set of possible outcomes of a strategy profile s^I is the set of plays that follow s^I .

Winning conditions and colorings are defined as usual. A profile of strategies $s = (s^0, \dots, s^{n-1})$ is winning if all its outcomes winning. Notice that the notion of winning does not regard one player, but the coalition of all players. In the context of distributed games, we are interested in strategy profiles which allow the grand coalition to coordinate towards ensuring a win against the environment.

2.1 GAMES WITH SEVERAL PLAYERS AND SIMULTANEOUS MOVES

We are particularly interested in strategies implemented by strategy automata. Defined as in the previous section, a strategy automaton for Player i is a structure $\mathcal{M}^i = (M, B^i, A^i, m_0, \mu, \sigma)$ that inputs observations from B^i as input and outputs actions from A^i .

As we focus on impossibility results, we choose examples of games with possibly simple winning conditions. A (*simple*) *safety game* is a game with two colours assigned by $\gamma : V \rightarrow \{\text{safe}, \#\}$ that are observable to all players, describing the winning condition $W = \{\text{safe}^\omega\}$. Hence, a play is winning if, and only if, it avoids the positions coloured by $\#$. For simplicity, we assume that the symbol $\#$ belongs to the observation set of every player and that $\beta^i(v) = \#$ for a player i precisely when $\gamma(v) = \#$.

Questions. Given a safety game, we ask the following questions:

- *Winner determination:* does there exist a winning strategy for the coalition of all players ?
- *Finite-state distributed winning:* Does there exist a distributed finite-state winning strategy, that is, a winning strategy profile s where s^i can be implemented by a finite-state automaton, for every player i ?

Proposition 3. *There are finite safety games with two players that require infinite memory.*

To see an example of such a game, consider the graph depicted in Figure 1. Here, Player 1 and Player 2 play against the environment with actions in $A^1 = A^2 = \{\circ, \times\}$, and receive observations $B^1 = B^2 = \{\bullet, \neg, \#\}$. All moves that are not represented in the picture are meant to lead to a sink position (not represented either) where both players receive the fatal observation $\#$ and lose; the observation \neg is left void in the picture.

The plays have a similar scenario for both players: they need to perform action \circ until receiving the observation \bullet , then, perform a number of \times action, and finally switch back to \circ , forever. (The option of playing \times will be ruled out soon.) Thus, for each player, the question is how many \times actions to perform after observing \bullet . Accordingly, the strategies available to player i correspond to functions $f^i : \omega \rightarrow \omega$ with $f^i(n) = m$ meaning, that if \bullet occurs after n steps, play \times for precisely m rounds.

The game graph gives us even more clues about which choices the players could make. One important detail is that Player 2 receives \bullet either in the same round as Player 1, or in the subsequent one. In the former case, the two players need to perform the same number of \times actions. Hence, any safe strategy must satisfy $f^1(n) = f^2(n)$, for all n . In the latter case, Player 2 must produce one more \times -action than Player 1, implying $f^2(n+1) = f^1(n) + 1$, for all n . Finally, we can observe that, if Player 2 receives \bullet in the first round, his only choice is to produce one \times . Hence, $f^2(0) = 0$. But to satisfy all these constraints, there is only one pair of strategies that can be safely chosen $f^i(n) = n$. Thus, this unique winning strategy for the coalition needs unbounded memory.

2.1 GAMES WITH SEVERAL PLAYERS AND SIMULTANEOUS MOVES

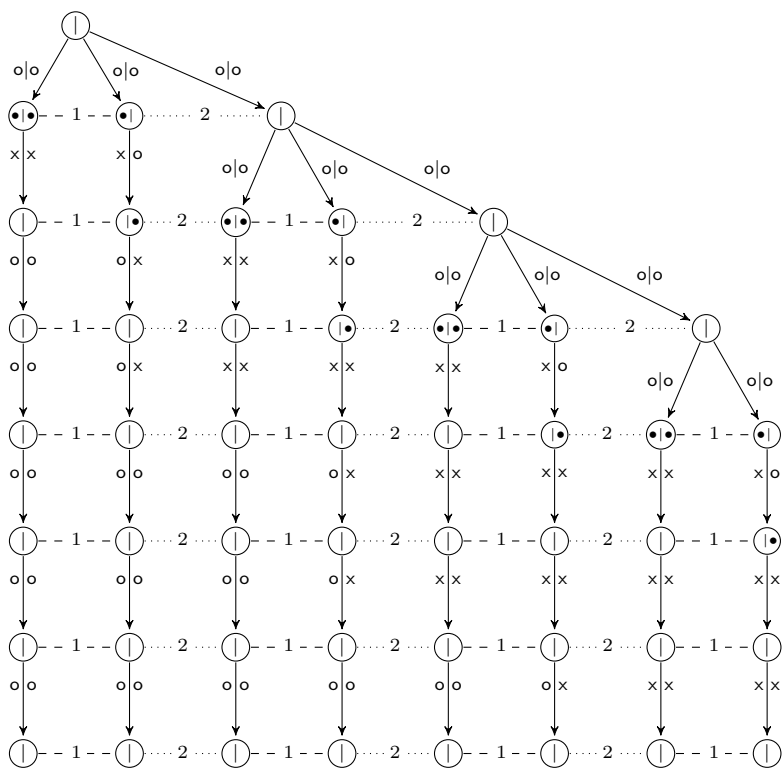


Figure 2: Extensive form of the game in Figure 1 (excerpt)

2.1 GAMES WITH SEVERAL PLAYERS AND SIMULTANEOUS MOVES

state, and the length of $|w|$ indicates the current head position with contents c . Accordingly, we choose $\Sigma \cup Q$ as an action alphabet for both players, and $\{\bullet, \neg, \#\}$ as the observation alphabet as before. The game graph features a test that the players produce the fixed initial configuration with empty tape when the play starts with a \bullet -observation for both players. All later simultaneous \bullet -observations lead to a testing that both players produce the same configurations and finally, if Player 2 receives the \bullet -observation after Player 1, we have a test that the configuration described of Player 2 succeeds the one described by Player 1, except when the current state is accepting in which case the test for equality applies.

In this scenario, the intended winning strategy for the coalition is the one that produces the n -th configuration of the machine \mathcal{M} upon observing \bullet in round n . This profile can be implemented with a finite-memory strategy, if and only if the machine \mathcal{M} halts. In that case, a simple diagonalisation argument shows that the amount of different memory states that need to be distinguished is not bounded by any computable function. Finally, the game admits a winning strategy, if and only if the machine does not halt on the empty tape – this problem is undecidable.

We can modify the above example to show that the question of whether the coalition has a winning strategy with *finite memory* is also undecidable. Towards this, it is sufficient to let the machine loop whenever it reaches a final configuration. Then, there exists a finite-memory winning strategy whenever the machine halts: produce the n -th configuration if the machine did not halt in n steps, otherwise produce the final configuration.

Corollary 5. *There exists a family of safety games for which the grand coalition has finite-state distributed winning strategies, but the amount of required memory grows faster than any computable function.*

To conclude, we remark that there are nevertheless interesting classes of games with imperfect information that can be solved algorithmically. In particular, this is the case when there is a hierarchic order among the players that reflects their observation ability in the sense that if a player j can distinguish between two positions v and v' then all players $i < j$ can do so: $\beta^i(v) = \beta^i(v')$ implies $\beta^j(v) = \beta^j(v')$ for all positions $v, v' \in V$ and every ordered pair of players $i < j$.