Winning Strategies Based on Testing Hypotheses in Games with Imperfect Information Evolving in Time.

Evgeny Dantsin
Roosevelt University, Chicago, USA

and

Sergei Soloviev
IRIT, University of Toulouse, France

One of main differences of algorithmic game theory from classical game theory is that many well known results (existence of equilibriums etc.) of game theory lead to algorithmically undecidable problems. (See, e.g. Rabin, M.O. 1957. Effective computability of winning strategies. Ann. Maths. Studies, 39, 147-157.)

The principal aim of this talk is to present an opposite situation when undecidability of certain algorithmic problem - in our case, the problem of identification of the strategy of an opponent - doesn’t prevent to win.

We present not an isolated case but a general framework that permits to describe large classes of games.

As it will be seen, there are connections with many other domains, e.g., inductive learning, but we shall concentrate in this talk mostly on the main construction of winning strategy.

Discrete-time dynamic games

Let $\mathcal{D}$ be a discrete-time dynamical system whose *state space* is a set S and whose time is the set $N = 0, 1, 2, \ldots$ of natural numbers. Let $T \subseteq S$ be a subset of states called *terminal* states. We are interested in whether $\mathcal{D}$ eventually reaches a terminal state. The system has two parameters a and b that may change over time and affect the behavior of $\mathcal{D}$.

We define a game for two players, called Alice and Bob, who control the parameters a and b respectively. Alice tries to bring $\mathcal{D}$ into a terminal state by choosing a value for her parameter a, and Bob chooses a value for his parameter b with the hope to avoid falling into T.

A *trajectory* of $\mathcal{D}$ is a sequence of states $s(0), s(1), \ldots$ where $s(t)$ is the state of the system at time instant t. Usually this sequence is uniquely determined by its initial state and a *transition rule*. When considering dynamical systems without parameters, it is common to define such a rule as a mapping that outputs a state $s(t+1)$ taking as input the previous "history" $s(0), \ldots, s(t)$. Since $\mathcal{D}$ has parameters, we have to change this definition.

At each time instant, Alice and Bob choose values for their parameters independently, not knowing the opponent's choice. Let A be a set of all possible values for Alice's parameter a. Her choice at time instant t is an element from A denoted by $\alpha(t)$. Similarly, B is a set of all possible values for Bob's parameter and $\beta(t) \in B$ is his choice at time instant t.Both **may** know and take into account all (or some) previous states and all (or some) previous values of parameters.

A *history* of $\mathcal{D}$ on time interval $[0, t]$ is a tuple $\gamma_t = \langle s_t, \alpha_t, \beta_t \rangle$ where

- s_t is a sequence $s(0), \ldots, s(t)$;
- α_t is a sequence $\alpha(0), \ldots, \alpha(t)$;
- β_t is a sequence $\beta(0), \ldots, \beta(t)$.

Thus, the history γ_t includes not only the trajectory on $[0, t]$ but also Alice's and Bob's choices on this time interval. For any set M, let M^n denote the set of all n-sequences of elements from M. Using this notation, we define a transition rule as follows.

A *transition rule* for $\mathcal{D}$ is a mapping τ that for any $t \in N$, outputs a state $s(t+1) \in S$ of the system at the time instant $t+1$ taking as input a tuple

$$\langle \gamma_t, \alpha(t+1), \beta(t+1) \rangle$$

where

- $\gamma_t \in S^t \times A^t \times B^t$ (a possible history on $[0, t]$);
- $\alpha(t+1) \in A$ (a value for Alice's parameter at the time instant $t+1$);
- $\beta(t+1) \in B$ (a value for Bob's parameter at the time instant $t+1$).

Such a rule τ uniquely determines a trajectory of $\mathcal{D}$ when given an initial state $s(0)$ and the values $\alpha(0), ..., \alpha(t), ...$ and $\beta(0), ..., \beta(t), ...$ for the parameters. It is possible to extend the definition of τ in order to include a constructing of an initial state into the rule: we assume that τ outputs $s(0)$ on the input $\langle \epsilon, \alpha(0), \beta(0) \rangle$ where ϵ is the empty sequence (empty history). It is possible also to define a trajectory beginning at any state.

Strategies Alice and Bob assign values to their parameters using *strategies*: any function $\alpha{:}N\times [A^t] \times [B^t] \times [S^t] \to A$ is called a *strategy for Alice* and, similarly, any function $\beta{:}N \times [A^t] \times [B^t] \times [S^t] \to B$ is a *strategy for Bob*. (The parameters in the square brackets are optional.) When strategies α and β are fixed, a trajectory $s(0), s(1), \ldots$ of $\mathcal{D}$ is completely determined by a transition rule τ. We say that Alice's strategy α *defeats* Bob's strategy β if there exists $t \in N$ such that $s(t)$ is a terminal state.

Our main interest is in cases when the strategies of Alice and Bob belong to certain sets and properties of these sets can be exploited.

Let $\mathcal{A}$ (resp. $\mathcal{B}$) be a set of functions from N to A (resp. B). Now we define a *discrete-time dynamic game* to be a tuple $\langle \tau, \mathcal{A}, \mathcal{B} \rangle$ where τ is a transition rule, $\mathcal{A}$ and $\mathcal{B}$ are sets of strategies that Alice and Bob can use. Finally, we say $\alpha \in \mathcal{A}$ is a *winning strategy* for Alice if α defeats each strategy $\beta \in \mathcal{B}$.

We call a strategy **uniformly winning** if it wins for every intial state.

Matching game Alice and Bob play a game that consists of a finite or infinite sequence of rounds. Before the first round, Alice has no money while Bob has m rubles. Alice does not know the value of m. In each round, they write down natural numbers (not knowing the opponent's number). Then the numbers are compared. If they coincide, Bob gives Alice one ruble. Otherwise Alice hands back all of the money won in the previous rounds (if any). Thus, as a result, either Bob has one ruble less or he gets all his m rubles back. If the ruble given to Alice is Bob's last ruble, Alice wins and they stop the game. Otherwise, they move on to the next round.

Via the framework this game may be seen as follows. Natural numbers written down by Alice and Bob are the values of their parameters: $A = B = N$. The state space S can be defined as the set N as well: $s(t) = m$ means that at time instant t (after comparing the written numbers) Bob has m rubles. Alice uses a strategy $\alpha{:}N \times N^t \times N^t \to N$ to choose her numbers: at time instant t she writes down the number $\alpha(t)$. Bob writes down his numbers using a function $\beta{:}N \times N^t \times N^t \times N \to N$, but his strategy is a pair (m_0, β) where m_0 is the number of rubles that he has before the first round and β is used for his choices.

The definition of transition rule τ has two cases:

$$s(0) = \begin{cases} m_0 - 1 & \text{if } \alpha(t+1) = \beta(t+1) \\ m_0 & \text{if } \alpha(t+1) \neq \beta(t+1) \end{cases}$$

for computing the initial state and

$$s(t+1) = \begin{cases} s(t) - 1 & \text{if } s(t) > 0 \text{ and } \alpha(t+1) = \beta(t+ \\ m_0 & \text{if } s(t) > 0 \text{ and } \alpha(t+1) \neq \beta(t+ \\ 0 & \text{if } s(t) = 0. \end{cases}$$

for computing further states.

We also need to specify $\mathcal{A}$ and $\mathcal{B}$. Assume, for example, that $\mathcal{A}$ is the set of all computable functions and $\mathcal{B}$ is the set of all primitive recursive functions. Does Alice have a winning strategy?

Toolbox for Alice

Hypotheses. Alice knows something about the set $\mathcal{B}$ of Bob's strategies but she does not know what strategy $\beta \in \mathcal{B}$ he actually uses. However, she can make *hypotheses* about Bob's actual strategy and then she can test them.

In the simplest case, Alice's hypotheses can be identified with Bob's strategies themselves. That is, Alice hypothesizes that Bob's actual strategy is a certain strategy $\beta \in \mathcal{A}$.

(Cf. **Matching Game** and pairs **(**$m_0 \in N$**, number of PR function** as hypotheses.)

In general Alice has a set H of *hypothesis*. A hypothesis $h \in H$ is a predicate on $\mathcal{B}$, i.e., a Boolean function from $\mathcal{B}$ to $\{\textbf{true}, \textbf{false}\}$. We assume that *Alice's hypotheses "cover" all Bob's choices*, which means that for any $\beta \in \mathcal{B}$, there exists $h \in H$ such that $h(\beta)$ is true. We also assume that *Alice can enumerate all her hypothesis*: the set H is enumerable.

Testers. Alice tests her hypotheses using a function called a *tester*.

A tester can be thought of as an oracle that answers Alice's queries. She issues a query when she wants to test her hypothesis about the actual choices of Bob. Suppose that, at time instant $t \geq 1$, Alice's hypothesis is $h \in H$. Then she issues a query that includes h and information about the behavior of the system on the previous time instants $\{0, 1, \ldots, t-1\}$. The oracle's answer is either 0 or 1, where 0 means "the hypothesis h is not refuted" and 1 means "the hypothesis h is refuted".

Formally, a tester $\mathcal{T}$ is a Boolean function such that

- the domain of $\mathcal{T}$ is the set of all 4-tuples $(h, t, \alpha|_t, \tau|_t)$, where
  - h is Alice's hypothesis;
  - $t \in N \setminus \{0\}$ is a time instant;
  - $\alpha|_t$ is a function from $\{0, 1, \ldots, t-1\}$ to A;
  - $\tau|_t$ is a function from $\{0, 1, \ldots, t-1\}$ to M^t, $\tau_t = s(0), ..., s(t-1)$.
- $\mathcal{T}$ is non-decreasing with respect to time t, i.e. if the hypothesis is refuted then it remains refuted.

Note that the tester has an implicit parameter. This parameter is Bob's actual strategy β. The

value $\mathcal{T}(h, t, \alpha|_t, \tau|_t)$ indeed depends on β because the observable trajectory depends on β.

Terminators. We assume that with every hypothesis h is associated a strategy α_h **winning uniformy** against every strategy β such that $h(\beta)$ holds. This function is given by an algorith called **find-terminator**.

(Cf. **Matching Game**)

Locality Condition. What may happen if the **tester** does not refute an hypothesis but the hypothesis is not right?

Why it may happen?

Because the difference of behaviour of β and β' is **not observable**.

If tester $\mathcal{T}_h$ is always true on the trajectory τ that corresponds to the strategies α, β then **there exists β' such that $h(\beta')$ and the trajectory that corresponds to α, β' is the same**.

Theorem .1 *Suppose that Alice has the algorithms* enumeration, tester, *and* find-terminator*. Suppose also that the locality condition is satisfied. Then Alice has an algorithm that, using these algorithms as subroutines, computes a winning strategy function for her.*

Proof- sketch. This strategy is constructed piecewise. Main idea: we define certain function that computes ”switching points” - the points where the strategy has to be changed. It happens when a hypothesis is to be refuted (is not in accord with the moves of Bob). If the hypothesis is not refuted (even if it not right) Alice will win sooner or later. (Here the locality condition is used.) Otherwise Alice will come to the right hypothesis and also will win. (...)

Rabin, M.O. 1957. Effective computability of winning strategies. Ann. Maths. Studies, 39, 147-157.