Perspective Games

Now you see me, now you don’t

Shubh Sharma, Siddhant Agarwal

2025-11-25

Chennai Mathematical Institute

Introduction

Perspective Games

Partial Information games : Games where players don't have all the information about the game arena and game state at all times.

Transverse Uncertainty : Sometimes player see all the information, some times they do not, like:

- Concurrent threads given control by a scheduler
- Clients connecting to a server, one at a time.
- Games with concurrent moves

Perspective Strategies or P-strategies are strategies, depending only on the part of the run that is within a player's domain, given by $\rho : V_1^{\oplus} \to V_1$.

Formal Description

Game Graph $G = \langle V_1, V_2, E, v_0, \mathrm{AP}, \tau \rangle$

- AP is a finite set of atomic propositions.
- $\tau : V_1 \sqcup V_2 \to 2^{\mathrm{AP}}$ assigns which atomic propositions are true in which locations.

Given a run $\rho = (\rho_1 \cdot \rho_2 \cdot \rho_3 \cdots) : V^\omega$, the computation of a run $\tau(\rho) : \left(2^{\mathrm{AP}}\right)^\omega$ is defined as

$$\tau(\rho) = \tau(\rho_1) \cdot \tau(\rho_2) \cdot \tau(\rho_3) \cdots$$

Winning condition L decides which runs are winning, given as one of the following:

- An LTL formula
- An ω-automata

Thus $L \subseteq \left(2^{\mathrm{AP}}\right)^\omega$.

Perspective Strategies

Given a game graph G, let the set of all finite runs on it be defined as $V^{\circledast}$.

Given a finite run ρ, we define $\pi_p : V^{\circledast} \to V_p^*$ as the projection map, which drops all vertices in the run not belonging to player p.

A Perspective strategy σ_p for a player p takes the part of a run restricted to player p vertices and decides a move based on that.

Thus, given a function $f : V_p^* \to V_p$, the following describes a P-strategy:

$$\sigma_p(\rho) :\equiv f(\ \pi_1(\rho)\ )$$

We consider games where each player can have a different kind of strategy,eg. in a PF-game, player 1 has a perspective strategy and player 2 has an ordinary (or full) strategy.

Deterministic Setting

We say that a strategy σ is winning for player 1, if for any strategy τ for player 2, the play induced by the strategies satisfies the winning condition.

Some nice theorems:

- Given a game $\mathcal{G}$
  - Player 1 FF-wins $\mathcal{G}$ iff Player 1 FP-wins $\mathcal{G}$
  - Player 1 PF-wins $\mathcal{G}$ iff Player 1 PP-wins $\mathcal{G}$
- There is a game $\mathcal{G}$ such that player 1 F-wins $\mathcal{G}$ but does not P-win $\mathcal{G}$.
- Perspective Games are not determined.

Probabalistic Setting

A probabalistic strategy for player i is a function $V^{\circledast}V_i \to \mathcal{D}(V)$, where $\mathcal{D}(V)$ is the set of probability distributions on V. Given strategies g_1 and g_2 for both players we define

$$\mathcal{P}_{g_1,g_2}(L) = [\![\rho \text{ is in } L \mid \rho \text{ is generated using } g_1, g_2]\!]$$

A strategy σ is almost winning for player 1 if for every strategy τ we have $\mathcal{P}_{g_1,g_2}(L) = 1$.

Some nice Theorems:

- There is a game $\mathcal{G}$ that is PF-almost winning for player 1, but not P-winning.
- There is a game $\mathcal{G}$ that is PP-almost winning but not PF-almost winning for player 1.
- Perspective games are not almost-determined.

Deterministic Setting Analysis

Results

Deciding whether Player 1 P-wins and finding a P-strategy in a perspective game $\langle G, \mathcal{U} \rangle$ is $\mathrm{EXPTIME}$-complete when $\mathcal{U}$ is a universal automata with a parity or a reachability winning condition. The problem can be solved in time polynomial in $|G|$ and exponential in $|\mathcal{U}|$.

Deciding whether Player 1 P-wins and finding a P-strategy in a perspective game $\langle G, \psi \rangle$ is $2\mathrm{EXPTIME}$-complete when ψ is an LTL specification. The problem can be solved in time polynomial in $|G|$ and doubly exponential in $|\psi|$.

Outline for Upper Bound

- Perform a polynomial conversion from winning condition $\mathcal{U}$ to an alternating tree automaton $\mathcal{A}_{\mathcal{G}}$.
- Show that finding a P-strategy for Player 1 is equivalent to finding an accepting run in $\mathcal{A}_{\mathcal{G}}$.
- Solving the emptiness problem for $\mathcal{A}_{\mathcal{G}}$ gives an upper bound for finding Player 1 P-strategy.

Tree Automata

A Tree Automata is given by the following tuple

$$\mathcal{A} = \langle \Sigma, Q, q_{\text{in}}, \delta, \alpha \rangle$$

where

- $\alpha \subseteq Q^\omega$ is the winning condition.
- $\delta : Q \times \Sigma \to Q^*$ gives the list of locations corresponding to children of the tree.
- Σ is set of labels for the nodes of the tree.

A run of the automata on a tree T can be through as wapping the tree along the edges of the automata. It can be given as a function $r : T \to Q$ such that.

- $r(\text{root}_T) = q_{\text{in}}$
- If v is a node labelled by a with children $v_1, v_2 ... v_n$ such that $r(v) = q$. Let $\delta(q, a) = q_1, q_2 ... q_n$ we have $r(v_i) = q_i$.

Add Diagram

Non-deterministic and Universal tree automata transitions return a list of words. Both of these can be combined and represented uniquely by descriving an alternating tree automata where the transitions have the type:

$$\delta : Q \times \Sigma \to \mathcal{B}(Q \times D)$$

Where $\mathcal{B}(Q \times D)$ are positive boolean formulas. Note that runs of ATAs can be over more than 1 trees. One would pick a set nodes, and directions for edge labels.

Upper Bound (Parity Automata)

We can think of a strategy as a tree. For player 1, if a play is in V_1 we choose one vertex as its successor. If the play is in V_2, then the control can go back to player 1 in multiple ways. This is how we interpret a strategy as a tree.

Thus, for our tree automata, we will simulate strategies on the game:

- $Q' = V \times Q \times [1..n]$
- $q_{0'} = \langle v_0, q_0, 0 \rangle$

If the game was in configuration (v, q) such that $v \in V_1$ and player 1 chooses to go to v'.

- Let $S^{v'}_{v,q}$ be $\bot$ if from $v' \in V_2$ player 2 can force the game in v_2 and win.
- Let $S^{v'}_{v,q}$ be all such (v'', q', i) where player 2 can take the game from v' to v'', q' is the vertex in the word-automata reached after following the path, and i is the highest priority seen along the path.

We now define the edges to be:

- From (v, q, i) if $S^{v'}_{v,q} = \bot$ or there is no edge from v to v' the transision is (v, q, i) on reading v' goes to fail.
- Otherwise (v, q, i) on reading v' goes to all vertices (v'', q', i') in $S^{v'}_{v,q}$ for vertex v'', if the set is empty, we accept the run.

Our winning condition is again parity, an for any vertex (v, q, i) we define its parity to be i.

We then have the following theorem:

> Given a game $\mathcal{G} = \langle G, \mathcal{U} \rangle$ where $\mathcal{U}$ is a universal parity tree game, we can convert it to a equivalent non-deterministic Rabin tree game and find a witness in time polynomial to $|G|$ and exponential to $|\mathcal{U}|$. Furthermore, we can make a transducer to extract a witness.

Lower Bound (Reachability)

We show a reduction from membership for linear bounded alternating turing machines. We assuming the automata alternates between existential and universal nodes.

Given such a machine M along with a word w, we know the size of the tape required, so we know the size of the configuration. Both players will take turns picking transitions of the from $Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ that the machine will take. Player 1 will have a winning strategy iff M accepts w.

Since the number of configurations is exponential in size of the word, we will only keep track of one specific position in the tape during the game.

Since the entire information of the tape is not kept track of, both players need simulate the turing machine. We punish player 1 if the simulation is not faithful, and we give her the power to force player 2 to play correctly.

The game proceeds by:

- Player 2 selecting a cell of the tape on which transitions will be verified.
- Player 1 does not know what cell player 2 selected, so she must play safely by remembering the tape content correctly.
- Player 1 takes her turn in 2 steps
  - she picks a transition $q_{\text{in}}, \gamma \to q', \gamma', d$ and claims tape symbol was γ.
  - She claims that the tape content under the head now, is λ.
- For player 2, the machine is in state q' and player 1 said the current tape letter is λ, so she must take a transition of the form $q', \lambda \to q'', \gamma'', d'$
- This process repeats with player 1 taking a transition from whatever state the turing machine is in.
- The run is accepted if the machine reaches the accept state.

This is a polynomial time reduction, this we have proved the $\mathrm{EXPTIME}$-hardness.

LTL Winning Condition

For a game $\mathcal{G} = \langle G, \psi \rangle$

- Build Buchi Automata that satisfies $\neg\psi$. Construction is exponential.
- Dualize to get a universal co-Buchi Automata, which is also a parity automata.
- Use the previous algorith, which is exponential in the size of the automaton.

This proves the upper bound.

The proof for the lower bound reduces the realizablity problem for an LTL formula ψ to a game $\mathcal{G} = (G, \psi)$. Finding an F strategy is known to be 2-$\mathrm{EXPTIME}$-complete.

The states of G are 2 copies of 2^{AP}, one for player 1 and 1 for player 2, and all transitions are of the from V_1 to V_2 or V_2 to V_1. Here, finding a P-strategy is equivalent to finding an F-startegy as vertices alternate.

Perspective ATL^* Model Checking

Perspective ATL^*

ATL^* is an extension of the logic CTL^* which captures the existence of strategies in a game. Perspective-ATL^* lets us quantify over perspective startegies.

There are 2 types of formulas

- State formulas φ

$$\varphi ::= \mathrm{AP} \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\langle S \rangle\rangle \psi \mid \langle\langle S \rangle\rangle_p \psi$$

- And there are path formulas ψ

$$\psi ::= \varphi \mid \neg\psi \mid \psi \vee \psi \mid \bigcirc \psi \mid \psi \mathcal{U} \psi$$

There is also the logic ATL (similarly perspective-ATL), which simplifies ATL^* by forcing all path operators to be preceeded by path quantifiers. eg $\langle\langle 1 \rangle\rangle \bigcirc \bigcirc p$ is not allowed.

Model Checking

Model Checking is the problem of verifying if a given model M satisfies a given formula φ.

The model checking problem for $\mathrm{perspective\text{-}ATL}^*$ is $\mathrm{2\text{-}EXPTIME\text{-}complete}$. The model checking problem for $\mathrm{perspective\text{-}ATL}$ is $\mathrm{PTIME\text{-}complete}$.

Conclusion

Probabalistic Setting

Structural Winning Condition

Memoryless Strategies

Thank You!

Thank you for attending our presentation. If you are interested in reading the paper, it can be found here:

https://dl.acm.org/doi/10.1145/3627705