Game Theory Project

Juan Pablo Bello Gonzalez

September 2024

1 introduction

extensive form games to model imperfect information games. They involve sequential decisions when players choose actions at differnt times (Prof Bryce Extensive Games). To represent these interactions we can use the extensive form gametree which lets us encode the timing information of a sequential game "the key idea is tht if player are making decisions at different times, then each point where a player makes a decision is represented as a node in a tree. We'll label the node with the player who is making the decision at that point. Also each level on the tree alternates players. And decisions avaliable to a player when they are making a decision will result in branches steming from that node. Those branches may lead to other decision nodes for various players in the game or they may lead to a terminal outcome represented by a leaf in the tree where as usual we will have a payoff for each of the players, in the case of poker and Khun poker this payoff will be zero sum. "
these levels of the tree correspond to a player making a decision and then player 2 reacting to that decision. This is a computationally intensive method (do poker tree start) even after symmetries removed.
How strategies and equillibria relate to this model? - We think of a strategy in an extensive form game as a complete contingent plan of actin that a player would do at any of their decision nodes, given the history wich is visible to them. So a strategy for player i takes as input any of the decision nodes for that player and produces an action that the player would play at that possition.
Counter factual regret minimization.
How this can be used to model poker and weakly solve poker

1.0.1 Modeling imperfect information

"To model imperfect information, the histories of each player d are partitioned into a collection d of so-called information sets. Each information set d groups together histories that Player i cannot distinguish between when he or she acts (MIT LECTUREs)" For instance, in the case of Kuhn poker, An information set of player 1 is JQ, JK which occurs after the draw and before any betting, at this point in time, Player 1 only has knowelde of what eh drew, J in this case. But his information set is bigger since, from his poov, P1 cannot know if P2 has Q or K.

2 Defining the game

According to (REGRET MINIMIZATION..) A formal definition of an extensive game includes the following:

- A finite set of two players $N = P_1, P_2$
- A finite set of ordered action histories H, such that the empty sequence is in H and every prefix of a sequence in H is also in H.
  - Denote $Z \subset H$ the terminal histories, corresponsding to leaves on the tree representation, after which the game, it is no one's turn, and players receive utilities.
  - Denote $A(h) = \{a : (h, a) \in H \setminus Z\}$ as the actions available for the player whose turn it is after a non-terminal history $h\ inH \setminus Z$.
- fdede

  In the case of Khun poker, there are a two distinct action sets, those corresponding to chance draw $A_1 = KJ, KQ, QJ, QK, JQ, JK$ with the ordering encoding the player ie KJ

  In the case of Khun poker H = set of ordered subsets of $C \times A \times A$ where $C = (a, b) : a \in Cardsb \in Cardsa \neq b$ Some examples (Q, J, b, b) $Z \subset H$ is the set of terminal histories. for instance (QJ, b, b), (QJ, p), etc. $A(h)$ are actions avaliable after a non-terminal history $h \in H$.
- A function $P : H \setminus Z \to N$ that determines the player whose turn it is to act at the end of a given history.

  EXPLANATION: In the Khun poker definition, $P([(c_1, c_2)]) = P1$ encodes that after the draw from chance, it is player 1's turn to act. $P([(c_1, c_2), b]) = P2$ encodes that after the draw from change and player 1's betting action, it is the turn of player 2.
- A function f_c. which assignes a probability distribution over at every decision point, ie at node corresponding to player i's decision, given the history the probabiity that they go transition to the avaliable options. This is just the definition of that distribution, we would like to find the optimal such distribution which maximizes utility?

  For each player $i \in N$, a partition

2.0.1 A strategy in an extensive form game

'Loosely speaking, a strategy describes how to act in every single possible situation. It is a recipe for how to act in entire game, a lookup table you can read your actions from. (INT8)'
For player i, a strategy should provide a probability sdistribution over actions avaliable to him at every information set where he is expected to perform an action, so not including terminal nodes. For khun poker:

2.0.2 Nash Equillibrium

By Nash's Theorem, any finite game is guaratneed to have a, possibly mixed, NE. This includes poker and Khun poker.

2.0.3 Information sets

For defining imperfect information sets, players do not have knoweldge of the entire state, player's perspectives are called information sets. "Information set is a set of game states (game tree nodes) that are not distinguishable for a player."

2.0.4 No Regret Learning

regret is central to **online learning**, which deals with repeatedly making decisions in imperfect information environemtns.
We must index by time, this means we get the right answers and can evaluate expert predictions.
Assume we have expert predictions. We wish to distribute trust. After the environemnt reacts, it will reveal the real outcome (sequential/online structure) and so we obtain a loss vector that evaluates out N experts.
"Regret at time t is a difference between our algorithm total loss and the best single expert loss"

2.0.5 Regret matching as an example of no-regret learning algorithm

EXPERTS are actions (ie you shuold fold, call, bet) WE can compute cummulatve regret wrt an expert i at time t, expressing how we regret not listening to particular epert i. and we update that experts weight.

2.0.6 connecting no-regret learning and game theory

This regret matching algorothm can be used to model something that evolves with time, but the issue with games is that we would like to input the game state/history at some point into the regret,
Since regret goes from costs to utility, we now wish to maximize it?
Both players in fact will adapt to one another via no-regret learning. So how do we keep track of the regret for each plater?

THEOREM: If two no-regret algorithms play a zero-sum game against one another for T iterations with average regret less than ϵ, then their average strategies approximate Nash Equilibrium (up to 2ϵ)

2.0.7 Average Overall Regret