Computing Solution Concepts of Normal-Form Games

Song Chong
EE, KAIST
songchong@kaist.edu
Computing Nash Equilibria of Two-Player, Zero-Sum Games

- Can be expressed as a linear program (LP), which means that equilibria can be computed in polynomial time
- Let $U_i^*$ be the expected utility for player $i$ in equilibrium (the value of the game). $U_i^* = -U_2^*$.
- The minimax theorem tells us that $U_1^*$ holds constant in all equilibria and that it is the same as the value that player 1 achieves under a minmax strategy by player 2
- Using this result, construct an LP (player 2’s program) that gives us player 2’s mixed strategy in equilibrium

\[
\begin{align*}
\text{minimize} & \quad U_1^* \\
\text{subject to} & \quad \sum_{k \in A_2} u_1(a_1^j, a_2^k) \cdot s_2^k \leq U_1^* & \forall j \in A_1 \\
& \quad \sum_{k \in A_2} s_2^k = 1 \\
& \quad s_2^k \geq 0 & \forall k \in A_2
\end{align*}
\]
Computing Nash Equilibria of Two-Player, Zero-Sum Games

• In the same fashion, construct an LP (player 1’s program) to give us player 1’s mixed strategies. This program reverses the roles of player 1 and player 2 in the constraints; the objective is to maximize $U_1^*$, as player 1 wants to maximize his own payoffs. This corresponds to the dual of player 2’s program.

$$\begin{align*}
\text{maximize} & \quad U_1^* \\
\text{subject to} & \quad \sum_{j \in A_1} u_1(a_1^j, a_2^k) \cdot s_1^j \geq U_1^* \quad \forall k \in A_2 \\
& \quad \sum_{j \in A_1} s_1^j = 1 \\
& \quad s_1^j \geq 0 \quad \forall j \in A_1
\end{align*}$$
Computing Nash Equilibria of Two-Player, Zero-Sum Games

• Finally, construct a formulation equivalent to our first linear program by introducing slack variables $r_1^j$ for every $j \in A_1$ and then replacing the inequality constraints with equality constraints.
Computing Nash Equilibria of Two-Player, General-Sum Games

• Unfortunately, the problem of finding a Nash equilibrium of a two-player, general-sum game cannot be formulated as a linear program. This is because the two players’ interests are no longer diametrically opposed. Thus, we cannot state our problem as an optimization problem.

• However, the problem of finding a Nash equilibrium of a two-player, general-sum game can be formulated as a linear complementarity problem (LCP). The LCP has no objective function at all, and is thus a constraint satisfaction problem, or a feasibility program, rather than an optimization problem.

\[
\begin{align*}
\sum_{k \in A_2} u_1(a_1^j, a_2^k) \cdot s_2^k + r_1^j &= U_1^* \quad \forall j \in A_1 \\
\sum_{j \in A_1} u_2(a_1^j, a_2^k) \cdot s_1^j + r_2^k &= U_2^* \quad \forall k \in A_2 \\
\sum_{j \in A_1} s_1^j &= 1, \quad \sum_{k \in A_2} s_2^k = 1 \\
s_1^j \geq 0, \quad s_2^k \geq 0 \quad \forall j \in A_1, \forall k \in A_2 \\
r_1^j \geq 0, \quad r_2^k \geq 0 \quad \forall j \in A_1, \forall k \in A_2 \\
r_1^j \cdot s_1^j = 0, \quad r_2^k \cdot s_2^k = 0 \quad \forall j \in A_1, \forall k \in A_2
\end{align*}
\]
Linear Complementarity Problem (LCP)

• We would still have a linear program if we do not include the last nonlinear constraint. However, we would also have a flaw in our formulation: $U_1^*$ and $U_2^*$ would be allowed to take unboundedly large values, because all of these constraints remain satisfied when both $U_i^*$ and $r_{ij}$ are increased by the same constant, for any given $i$ and $j$. This is why we add the nonlinear constraint, called the complementarity condition.

• Why does the complementarity condition fix our problem formulation?
  • The condition requires that the slack variable $r_{ij}$ is zero exactly when its corresponding action $a_{ij}$ is a best response to the mixed strategy $s_{-i}$.
  • So, the condition captures the fact that, in equilibrium, all strategies that are played with positive probability must yield the same expected payoff, while all strategies that lead to lower expected payoffs are not played. That is, each player plays a best response to the other player’s mixed strategy, which is the definition of a Nash equilibrium.

• The Lemke–Howson algorithm is the best-known algorithm designed to solve the LCP problem and is guaranteed to find a sample Nash equilibrium.

• **Theorem 4.2.1** The problem of finding a sample Nash equilibrium of a general-sum finite game with two or more players is *PPAD-complete* (a less familiar complexity class).

• PPAD-complete has shown to be neither NP-complete nor easier than NP-complete. However, it is generally believed that PPAD≠P and, in the worst case, it will take exponential time.
Beyond Sample NE Computation

• In two-player, general-sum games, instead of just searching for a sample equilibrium, we may want to find an equilibrium with a specific property as follows

1. **(Uniqueness)** Given a game $G$, does there exist a unique equilibrium in $G$?

2. **(Pareto optimality)** Given a game $G$, does there exist a strictly Pareto efficient equilibrium in $G$?

3. **(Guaranteed payoff)** Given a game $G$ and a value $v$, does there exist an equilibrium in $G$ in which some player $i$ obtains an expected payoff of at least $v$?

4. **(Guaranteed social welfare)** Given a game $G$, does there exist an equilibrium in which the sum of agents’ utilities is at least $k$?

5. **(Action inclusion)** Given a game $G$ and an action $a_i \in A_i$ for some player $i \in N$, does there exist an equilibrium of $G$ in which player $i$ plays action $a_i$ with strictly positive probability?

6. **(Action exclusion)** Given a game $G$ and an action $a_i \in A_i$ for some player $i \in N$, does there exist an equilibrium of $G$ in which player $i$ plays action $a_i$ with zero probability?
Beyond Sample NE Computation

• Unfortunately, all of these questions are hard in the worst case
  
  **Theorem 4.2.3** The following problems are NP-hard when applied to Nash equilibria: uniqueness, Pareto optimality, guaranteed payoff, guaranteed social welfare, action inclusion, and action exclusion

• We may also want to determine *all* Nash equilibria
  
  **Theorem 4.2.4** Computing all of the Nash equilibria of a two-player, general-sum game requires worst-case time that is exponential in the number of actions for each player
Computing Nash Equilibria of $n$-Player, General-Sum Games

- For $n$-player games where $n \geq 3$, the problem of finding a Nash equilibrium can no longer be represented even as an LCP.
- While it does allow a formulation as a *nonlinear complementarity problem*, such problems are often hopelessly impractical to solve exactly.
In a two-player, general-sum game, a maxmin strategy for player $i$ is a strategy that maximizes his worst-case payoff, presuming that the other player $j$ follows the strategy that will cause the greatest harm to $i$.

A minmax strategy for $j$ against $i$ is such a maximum-harm strategy.

Maxmin and minmax strategies can be computed in polynomial time because they correspond to Nash equilibrium strategies in related zero-sum games.

How to compute a maxmin strategy for player 1?

- Let $G$ be an arbitrary two-player game $G = (\{1, 2\}, A_1 \times A_2, (u_1, u_2))$ and define a related zero-sum game $G' = (\{1, 2\}, A_1 \times A_2, (u_1, -u_1))$.
- By the minmax theorem, every strategy for player 1 which is part of a Nash equilibrium strategy profile for $G'$ is a maxmin strategy for player 1 in $G'$.
- By definition, player 1's maxmin strategy is independent of player 2's utility function. Thus, player 1's maxmin strategy is the same in $G$ and in $G'$.
- Problem of finding a maxmin strategy in $G$ thus reduces to finding a Nash equilibrium of $G'$, which can be solved by an LP.

The computation of minmax strategies follows the same way.
Identifying Dominated Strategies

- Iterated removal of strictly dominated strategies
  - The same set of strategies will survive regardless of the elimination order, and all Nash equilibria of the original game will be contained in this set
  - Thus, this method can be used to narrow down the set of strategies to consider before attempting to identify a sample Nash equilibrium
  - In the worst case this procedure will have no effect—many games have no dominated strategies. In practice, however, it can make a big difference

- Iterated removal of weakly or very weakly dominated strategies
  - The set of strategies that survive iterated removal can differ depending on the order in which dominated strategies are removed. As a consequence, removing weakly or very weakly dominated strategies can eliminate some equilibria of the original game
  - However, since no new equilibria are ever created by this elimination and every game has at least one equilibrium, at least one of the original equilibria always survives. This is enough if all we want to do is to identify a sample Nash equilibrium
  - Furthermore, iterative removal of weakly or very weakly dominated strategies can eliminate a larger set of strategies than iterative removal of strictly dominated strategies and so will often produce a smaller game

- The complexity of determining whether a given strategy can be removed depends on
  - Domination by a pure or mixed strategies
  - Strict, weak or very weak domination
  - Only domination or survival under iterated removal of dominated strategies
Domination by a Pure Strategy

• Algorithm for determining whether $s_i$ is strictly dominated by any pure strategy

```
forall pure strategies $a_i \in A_i$ for player $i$ where $a_i \neq s_i$ do
  dom ← true
  forall pure-strategy profiles $a_{-i} \in A_{-i}$ for the players other than $i$ do
    if $u_i(s_i, a_{-i}) \geq u_i(a_i, a_{-i})$ then
      dom ← false
      break
  if dom = true then
    return true
return false
```

• Need to check every pure strategy $a_i$ for player $i$ and every pure-strategy profile $a_{-i}$ for the other players, meaning that we do not need to check every mixed strategy profile of the other players

• This is because if $u_i(s_i, a_{-i}) < u_i(a_i, a_{-i})$ for every pure-strategy profile $a_{-i} \in A_{-i}$, then there cannot exist any mixed-strategy profile $s_{-i} \in S_{-i}$ for which $u_i(s_i, s_{-i}) \geq u_i(a_i, s_{-i})$ due to the linearity of expectation

• The case of weak or very weak dominance can also be tested using essentially the same algorithm with some modification

• For all of the definitions of domination, the complexity of the procedure is $O(|A|)$, linear in the size of the normal-form game
Domination by a Mixed Strategy

• Cannot use a simple algorithm like the one used to test domination by a pure strategy because mixed strategies cannot be enumerated. However, it turns out that we can still answer the question in polynomial time by solving a linear program.

• Without loss of generality, assume that player $i$’s utilities are strictly positive.

• A feasibility program to describe strict domination by a mixed strategy:

$$\sum_{j \in A_i} p_j u_i(a_j, a_{-i}) > u_i(s_i, a_{-i}) \quad \forall a_{-i} \in A_{-i}$$

$$p_j \geq 0 \quad \forall j \in A_i$$

$$\sum_{j \in A_i} p_j = 1$$

• This formulation, however, does not constitute a linear program because the constraints in linear programs must be weak inequalities.
Domination by a Mixed Strategy

• Instead, we must use the LP that follows

\[
\begin{align*}
\text{minimize} & \quad \sum_{j \in A_i} p_j \\
\text{subject to} & \quad \sum_{j \in A_i} p_j u_i(a_j, a_{-i}) \geq u_i(s_i, a_{-i}) \quad \forall a_{-i} \in A_{-i} \\
& \quad p_j \geq 0 \quad \forall j \in A_i
\end{align*}
\]

• This linear program simulates the strict inequality of previous formulation through the objective function
  • Because no constraints restrict the \( p_j \)'s from above, this LP will always be feasible. However, in the optimal solution the \( p_j \)'s may not sum to 1
  • In the optimal solution, we will have \( \sum_{j \in A_i} p_j u_i(a_j, a_{-i}) = u_i(s_i, a_{-i}) \) for at least some \( a_{-i} \in A_{-i} \)
  • A strictly dominating mixed strategy therefore exists if and only if the optimal solution to the LP has objective function value strictly less than 1
  • In this case, we can add a positive amount to each \( p_j \) in order to cause the weak inequality constraint to hold in its strict version everywhere while achieving the condition \( \sum_j p_j = 1 \)
A feasibility program to describe very weak domination by a mixed strategy

\[ \sum_{j \in A_i} p_j u_i(a_j, a_{-i}) \geq u_i(s_i, a_{-i}) \quad \forall a_{-i} \in A_{-i} \]

\[ p_j \geq 0 \quad \forall j \in A_i \]

\[ \sum_{j \in A_i} p_j = 1 \]

An LP to check weak domination by a mixed strategy can be derived by adding an objective function to the above feasibility program.

\[
\text{maximize} \quad \sum_{a_{-i} \in A_{-i}} \left[ \left( \sum_{j \in A_i} p_j u_i(a_j, a_{-i}) \right) - u_i(s_i, a_{-i}) \right] \\
\text{subject to} \quad \sum_{j \in A_i} p_j u_i(a_j, a_{-i}) \geq u_i(s_i, a_{-i}) \quad \forall a_{-i} \in A_{-i} \tag{4.48} \\
p_j \geq 0 \quad \forall j \in A_i \tag{4.49} \\
\sum_{j \in A_i} p_j = 1 \tag{4.50} \]
Domination by a Mixed Strategy

• Because of constraint (4.49), any feasible solution will have a nonnegative objective value.

• If the optimal solution has a strictly positive objective, the mixed strategy given by the $p_j$'s achieves strictly positive expected utility for at least one $a_{-i} \in A_{-i}$, meaning that $s_i$ is weakly dominated by this mixed strategy.

• Observe that all of the linear programs above can be modified to check whether a strategy $s_i$ is strictly dominated by any mixed strategy that only places positive probability on some subset of $i$’s actions $T \subset A_i$. This can be achieved simply by replacing all occurrences of $A_i$ by $T$ in the linear programs.
We only consider pure strategies as candidates for removal; indeed, as it turns out, it never helps to remove dominated mixed strategies when performing iterated removal.

It is important, however, that we consider the possibility that pure strategies may be dominated by mixed strategies.

For all three flavors of domination, it requires only polynomial time to iteratively remove dominated strategies until the game has been maximally reduced.

A single step of this process consists of checking whether a pure strategy of a player is dominated by any other mixed strategy, which can be answered in polynomial time by solving a linear program.

Each step removes one pure strategy for one player, so there can be at most \( \sum_{i \in N} |A_i| \) steps.
Iterated Dominance

• Computational questions regarding *which strategies remain in reduced games*

  1. **(Strategy elimination)** Does there exist some elimination path under which the strategy $s_i$ is eliminated?

  2. **(Reduction identity)** Given action subsets $A_i' \subseteq A_i$ for each player $i$, does there exist a maximally reduced game where each player has the actions in $A_i'$?

  3. **(Reduction size)** Given constants $k_i$ for each player $i$, does there exist a maximally reduced game where each player $i$ has exactly $k_i$ actions

• **Theorem 4.5.1** For iterated strict dominance, the strategy elimination, reduction identity and reduction size problems are in P. For iterated weak dominance, these problems are NP-complete