Algorithms for finding Nash Equilibria

Ethan Kim

School of Computer Science
McGill University

Outline

1. Definition of bimatrix games
2. Simplifications
3. Setting up polytopes
4. Lemke-Howson algorithm
5. Lifting simplifications

Bimatrix Games

- Given a bimatrix game (A, B) with $m \times n$ payoff matrices A and B, a **mixed strategy** for player 1 is a vector $x \in \mathcal{R}^m$ with nonnegative components that sum to 1. For player 2, a mixed strategy is a vector $y \in \mathcal{R}^n$.
- The **support** of a mixed strategy is the set of pure strategies that have positive probability. A **best response** to y is a mixed strategy x that maximizes the expected payof $x^T Ay$, and vice versa. A **Nash equilibrium** is a pair of mutual best responses.

Best Response Condition

Lemma

A mixed strategy x is a best response to a mixed strategy y if and only if all pure strategies in its support are pure best responses to y (And vice versa).

Proof.

Let $(Ay)_i$ be the ith component of Ay, which is the expected payoff to player 1 when playing row i. Let $u = max_i(Ay)_i$. Then,

$$x^T Ay = \sum_i x_i(Ay)_i = \sum_i x_i(u-(u-(Ay)_i)) = u-\sum_i x_i(u-(Ay)_i).$$

Since the sum $\sum_i x_i(u-(Ay)_i)$ is nonnegative (for $x_i \geq 0$, $u-(Ay)_i \geq 0$), $x^T Ay \leq u$. The expected payoff $x^T Ay$ achieves the maximum u iff that sum is 0. So if $x_i > 0$, then $(Ay)_i = u$. □

Some simplifications..

- **Symmetry assumption:**
  We first assume that the game is *symmetric*. So the payoff matrix C is an $n \times n$ matrix $C = A = B^T$.

Some simplifications..

- **Symmetry assumption:**
  We first assume that the game is *symmetric*. So the payoff matrix C is an $n \times n$ matrix $C = A = B^T$.
- **Nondegeneracy assumption:**
  A bimatrix game is **nondegenerate** if the $\#$ of pure best responses to any mixed strategy never exceeds the size of its support.
  $\rightarrow$ the submatrices induced by the supports are full-rank.

Some simplifications..

- **Symmetry assumption:**
  We first assume that the game is *symmetric*. So the payoff matrix C is an $n \times n$ matrix $C = A = B^T$.
- **Nondegeneracy assumption:**
  A bimatrix game is **nondegenerate** if the $\#$ of pure best responses to any mixed strategy never exceeds the size of its support.
  $\rightarrow$ the submatrices induced by the supports are full-rank.
- So in a symmetric, nondegenerate game, a NE has support size equal to the $\#$ of pure best responses.

An Example of Symmetric Games

Consider the payoff matrices:

$$C = \begin{pmatrix} 0 & 3 & 0 \\ 0 & 0 & 3 \\ 2 & 2 & 2 \end{pmatrix} = A = B^T$$

Best Response Condition gives a polyhedron..

- By the Best Response Condition, an equilibrium is given if any pure strategy is either a best response (to a mixed strategy) or is played with probability 0.

Best Response Condition gives a polyhedron..

- By the Best Response Condition, an equilibrium is given if any pure strategy is either a best response (to a mixed strategy) or is played with probability 0.
- This can be captured by **polytopes** whose facets represent pure strategies, either as best responses, or having probability zero.

Best Response Polyhedron

- Define the maximum expected payoff for a strategy x_k for $k \in N$ as:
$$u = max\{(Ay)_k | k \in N\}$$
- A *best response polyhedron* of a player is the set of the player's mixed strategies with the *upper envelop* of expected payoffs to the *opponent*.
- E.g. For player 2, it is (y_4, y_5, y_6, u) that fulfill the following:

$$\begin{aligned} 0y_4 + 3y_5 + 0y_6 &\leq u \\ 0y_4 + 0y_5 + 3y_6 &\leq u \\ 2y_4 + 2y_5 + 2y_6 &\leq u \\ y_4, y_5, y_6 &\geq 0 \\ y_4 + y_5 + y_6 &= 1 \end{aligned}$$

Best Response Polyhedron

In general, the set of mixed strategies are represented by the polyhedron:

$$\overline{P} = \{(x, u) \in \mathcal{R}^N \times \mathcal{R} | x \geq \mathbf{0}, \mathbf{1}^T x = 1, C^T x \leq \mathbf{1} u\}$$

We can simplify this polyhedron, first by assuming:

- C is nonnegative and has no zero column.
- (We can do this by adding a constant to C)

Then, we will elimiate the payoff variable u.

From $\overline{P}$ to P..

- For $\overline{P}$, we divide each inequaility $\sum_{i \in N} c_{ij} x_i \leq u$ by u, which gives $\sum_{i \in N} c_{ij}(x_i/u) \leq 1$.
- Treat each $z_i = x_i/u$ as new variable, and call the resulting polyhedron P. We then have:

$$P = \{z \in \mathcal{R}^N | z \geq \mathbf{0}, C^T z \leq \mathbf{1}\}.$$

- In effect: (1) the expected payoffs u are normalized to 1, and (2) the conditions $\mathbf{1}^T x = 1$ are dropped.
- Non-zero vectors $z \in P$ are converted back to probability vectors by multiplying $u = \frac{1}{\sum_i z_i}$, and this scaling factor u is the expected payoff to the opponent.

From $\overline{P}$ to P..

- The set $\overline{P}$ is in 1-1 correspondence with $P - \{\mathbf{0}\}$ with the map $(x, u) \mapsto x \cdot (1/u)$. ("projective transformations")
- Since binding inequality in $\overline{P}$ corresponds to a binding inequality in P, the transformation preserves face incidences.

Best Response **Polytope**

- Because C is nonnegative & has no zero column, P is a bounded, fully dimensional polytope.
- Because of nondegeneracy assumption, P is *simple*, i.e. every vertex lies on exactly N facets of the polytope.
- A facet is obtained by making one of the inequalities *binding*, i.e. converting it to an equality.

Best Response Polytope

We say a strategy i is represented at a vertex z, if either $z_i = 0$, or $C_i z = 1$, or both (i.e. At least one of the two inequalities for strategy i is tight at z.). Then:

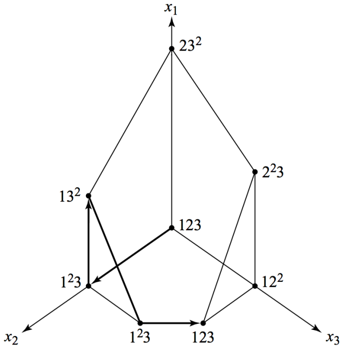
Theorem

If a vertex z represents all strategies, then either $z = \mathbf{0}$, or the corresponding (x, x) is a symmetric Nash.

Proof.

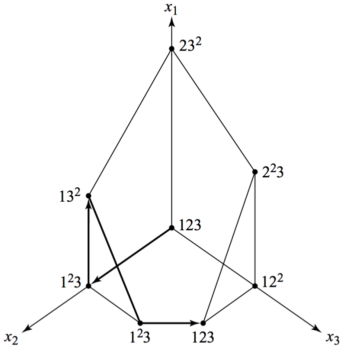
Assume $z \neq \mathbf{0}$. Then, the corresponding $x = u \cdot z$ is well defined, and x_i's are nonnegative numbers adding to 1. To see (x, x) is a Nash, observe that x satisfies the Best Response Condition: for every positive x_i's, $C_i z = 1$. Thus, every support is a best response. □

Lemke-Howson Algorithm



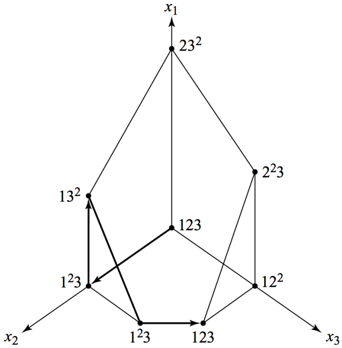
- Finds a vertex $z \neq \mathbf{0}$, where every strategy is represented.

Lemke-Howson Algorithm



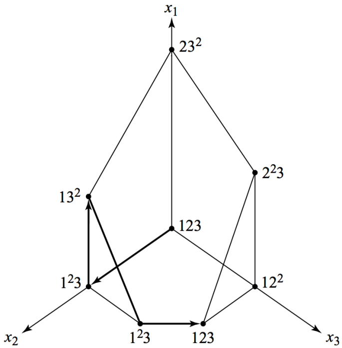
- Finds a vertex $z \neq \mathbf{0}$, where every strategy is represented.
- First, we label each facet of P by the strategy it represents: note that there are two facets (one for $(Cz)_i = 1$ and the other for $z_i = 0$).

Lemke-Howson Algorithm



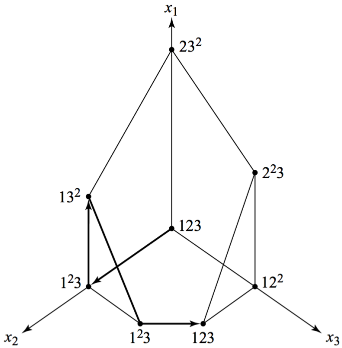
- Finds a vertex $z \neq \mathbf{0}$, where every strategy is represented.
- First, we label each facet of P by the strategy it represents: note that there are two facets (one for $(Cz)_i = 1$ and the other for $z_i = 0$).
- Then, label each *vertex* by the labels of adjacent facets.

Lemke-Howson Algorithm



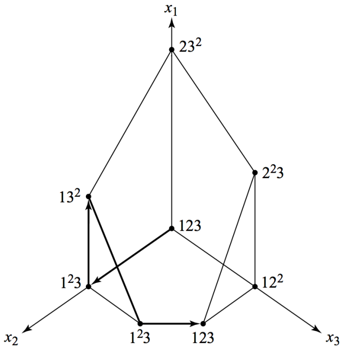
- Due to nondegeneracy, each vertex has precisely N adjacent facets, i.e. representing strategies.

Lemke-Howson Algorithm



- Due to nondegeneracy, each vertex has precisely N adjacent facets, i.e. representing strategies.
- $\rightarrow$ Each vertex has precisely N labels, while for each strategy i, both inequalities can be tight.

Lemke-Howson Algorithm



- Due to nondegeneracy, each vertex has precisely N adjacent facets, i.e. representing strategies.
- $\rightarrow$ Each vertex has precisely N labels, while for each strategy i, both inequalities can be tight.
- So a vertex can be labeled with duplicate copies of strategy i, while missing some other strategy j.

Lemke-Howson Algorithm

1. Set the starting vertex $v_0 = \mathbf{0}$. (This is a vertex of P.)

Lemke-Howson Algorithm

1. Set the starting vertex $v_0 = \mathbf{0}$. (This is a vertex of P.)
2. Choose an arbitrary strategy i, and <u>relax</u> the corresponding inequality. We are then taken to an adjacent vertex v_1. This vertex has $z_i \neq 0$ for the previously chosen strategy i.

Lemke-Howson Algorithm

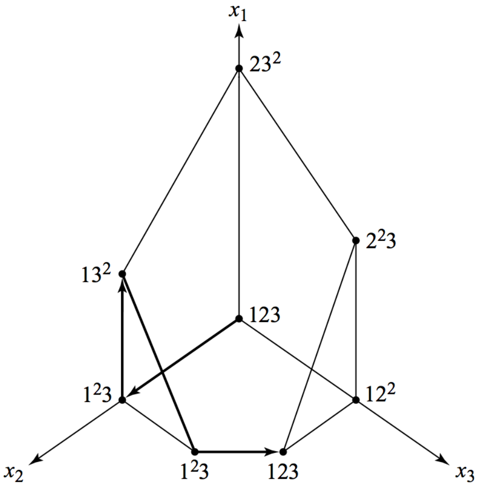
1. Set the starting vertex $v_0 = \mathbf{0}$. (This is a vertex of P.)
2. Choose an arbitrary strategy i, and relax the corresponding inequality. We are then taken to an adjacent vertex v_1. This vertex has $z_i \neq 0$ for the previously chosen strategy i.
3. At v_1, all strategies are represented except i, and one other strategy j is represented "twice" (i.e. both $z_j = 0$ and $(Cz)_j = 1$). By relaxing one of these two inequalities, we can reach two new vertices (one being v_0, and the other being v_2).

Lemke-Howson Algorithm

1. Set the starting vertex $v_0 = \mathbf{0}$. (This is a vertex of P.)
2. Choose an arbitrary strategy i, and relax the corresponding inequality. We are then taken to an adjacent vertex v_1. This vertex has $z_i \neq 0$ for the previously chosen strategy i.
3. At v_1, all strategies are represented except i, and one other strategy j is represented "twice" (i.e. both $z_j = 0$ and $(Cz)_j = 1$). By relaxing one of these two inequalities, we can reach two new vertices (one being v_0, and the other being v_2).
4. If v_2 again represents a strategy twice, repeat Step 3. Otherwise, we have reached a vertex that represents all strategies, each exactly once.

Lemke-Howson Algorithm

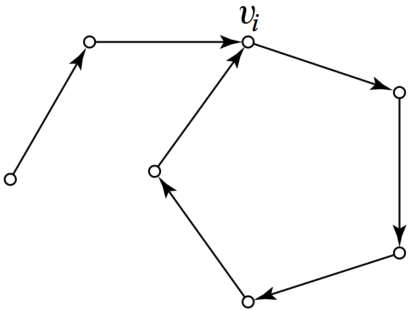
Going back to the example above..



Proof of Correctness

Why does the algorithm terminate?

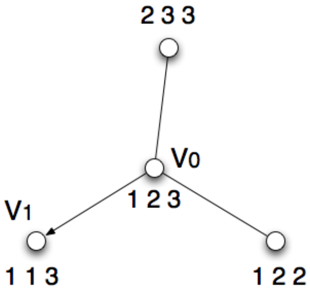
- **No internal vertex v_i can be revisited:**
  Repeating v_i would mean that there are 3 vertices adjacent to v_i that are reachable by relaxing a constraint with doubly represented strategy.



Proof of Correctness

Why does the algorithm terminate?

- **The initial vertex v_0 cannot be revisited:**
  Let i denote the strategy we initially relaxed to depart from v_0. Along the path, the algorithm never *picks up* strategy i until it terminates. But all vertices adjacent to v_0 represents strategy i, except v_1. Since v_1 cannot be revisited, v_0 cannot be revisited.



Proof of Correctness

Why does the algorithm terminate?

- **No internal vertex v_i can be revisited:**
- **The initial vertex v_0 cannot be revisited:**
- $\Rightarrow$ Note that P has a *finite* number of vertices. If a vertex represents a strategy twice, there is always a new vertex to reach, other than the one we came from. Therefore, LH algorithm finds a vertex represents *all* strategies.

Linear Complimentarity Problem(LCP)

- The polytope P doesn't provide us a NE; it simply gives us the set of mixed strategies.
- For a point $z \in P$ to be a NE, it needed to *represent* all strategies, i.e. all strategies with positive probabilities are best responses.
- This can be captured by the complimentarity condition:

$$z^T(\mathbf{1} - Cz) = 0$$

  , which is equivalent to $x_i = 0$ or $(Cx)_i = u$. By the BRC, this implies that x is a best response to itself.
- (See von Stengel 2002 for more detailed treatment of LCP.)

Edge Traversal

- Edge traversal between two vertices is implemeted algebraically by *pivoting* with variables entering and leaving a basis, while nonbasic variables represents the current facets. (Same as in Simplex algorithm!)
- The difference from Simplex algorithm is the rule for choosing the next entering variable: in Simplex Alg, the objective function dictates this choice. In LH algorithm, the *complementary* pivoting rule chooses the nonbasic variable with duplicate label to enter the basis.

Lifting Symmetry

- To handle non-symmetric bimatrix games, one can construct two polytopes P and Q, one for each player.

Lifting Symmetry

- To handle non-symmetric bimatrix games, one can construct two polytopes P and Q, one for each player.
- Each "move" in the algorithm can be achieved by finding a new vertex from the polytope P and Q in an alternating fashion.

Lifting Symmetry

- To handle non-symmetric bimatrix games, one can construct two polytopes P and Q, one for each player.
- Each "move" in the algorithm can be achieved by finding a new vertex from the polytope P and Q in an alternating fashion.
- In fact, this is a path on the product polytope $P \times Q$, given by the set of pairs (x, y) of $P \times Q$.

Lifting Symmetry

- To formulate a non-symmetric game into a symmetric game:

$$z = \begin{pmatrix} x \\ y \end{pmatrix}, \quad C = \begin{pmatrix} 0 & A \\ B^T & 0 \end{pmatrix}$$

- Then the normalization is done separately for x and y rather than the vector z as a whole.
- The edges in the product graph $P \times Q$ is then traversed alternatively.

Lifting Nondegeneracy

- The complementary path computed by LH is unique only if the leaving variable (dropping strategy) is unique. If not, then the system has degenerate basic feasible solutions, and LH algorithm may cycle unless the leaving variable is chosen in a systematic way.
- Degeneracy can be resolved by the standard lexicographic perturbation techniques from linear programming: (1) replace $B^T x \leq 1$ by $B^T x \leq 1 + (\epsilon, \ldots, \epsilon^n)$ (2) when choosing the leaving variable by pivoting rule, use the lexico-minimum rules.
- See von Stengel 2002 for a more detailed exposition.

Consequences of LH algorithm

- Lemke-Howson algorithm always finds a Nash equilibrium for any 2-player bimatrix games.
  $\Rightarrow$ Proof for existence of Nash, with an algorithm to find one.
- A nondegenerate bimatrix game has an **odd** number of Nash equilibria.
  Why? The LH algorithm can start at any Nash equilibrium, not just at $\mathbf{0}$. When LH is started at a NE not on the path starting from $\mathbf{0}$, it would terminate at another NE. Since there may be such disjoint paths with both endpoints being NE, there are odd $\#$ of NE (excluding the $\mathbf{0}$).

Concluding remarks..

- LH *finds* a NE in a finite number of steps, but how fast does it run?
  Savani & von Stengel (2006) gave a class of square bimatrix games for which LH algorithm takes an exponential number of steps in the dimension d of the game.
- What is the complexity of finding a Nash Equilibrium in a bimatrix game?
  The usual class of NP doesn't apply – there is always a NE!
  Daskalakis, Goldberg and Papadimitriou showed that it is *PPAD-Complete*. (in another lecture)

References

- Lemke and Howson, *Equilibrium points of bimatrix games*, SIAM Journal of Applied Mathematics, 12, pp413-423, 1964.
- Papadimitriou, Chapter 2 "The complexity of finding Nash Equilibria", *Algorithmic Game Theory*
- von Stengel, Chapter 3 "Equilibrium Computation for Two-Player Games", *Algorithmic Game Theory*
- Savani and von Stengel, *Hard-To-Solve Bimatrix Games*, Econometrica, Vol 74, No. 2 (March 2006)
- C. Daskalakis, P. Goldberg and C. Papadimitriou, *The Complexity of Computing a Nash Equilibrium*, to appear SIAM Journal on Computing.