

Computing Nash-equilibria in 2-player games and beyond

Algorithms Reading Group

Aditya Menon

June 3, 2007

Introduction

Nash equilibrium

A formal definition

Complexity

Characterizing complexity

PPAD

Finding equilibria in 2-player games

Lemke-Howson algorithm

Other approaches

Nash-equilibrium - Recap

- ▶ Refers to a special kind of state in an n -player game
- ▶ No player has an incentive to *unilaterally* deviate from his current strategy
 - ▶ A kind of “stable” solution
- ▶ Existence depends on the type of game
 - ▶ If strategies are “pure” i.e. deterministic, does not have to exist in the game
 - ▶ If strategies are “mixed” i.e. probabilistic, then it *always* exists
 - ▶ Yet how do we find it!?!

Notation

- ▶ Suppose that player p follows the mixed strategy $\mathbf{x}_p = (x_{p1}, \dots, x_{pn_p})$
 - ▶ The i th entry gives the probability that player p plays move i
- ▶ Let $\mathbf{x} := (\mathbf{x}_1, \dots, \mathbf{x}_n)$ be the collection of strategies for all players
- ▶ Let the function $U_p(\mathbf{x})$ denote the expected utility or payoff that player p gets when each player uses the strategy dictated in \mathbf{x}
 - ▶ Based on $u_p(s_1, \dots, s_n)$, which is the utility when player q plays s_q

Formal definition

- ▶ We say that $\mathbf{x}^* = (\mathbf{x}_1^*, \dots, \mathbf{x}_n^*)$ is a Nash equilibrium if...
 - ▶ “No player has an incentive to *unilaterally* deviate from his current strategy”
- ▶ If player p decides to switch to a strategy \mathbf{y}_p , then write the resulting strategy set as $\mathbf{x}_{-p}; \mathbf{y}_p$
- ▶ So, \mathbf{x}^* is a N.E. if, for *every* player p , and for *any* mixed strategy \mathbf{y}_p for that player, we have

$$U_p(\mathbf{x}^*) \geq U_p(\mathbf{x}_{-p}^*; \mathbf{y}_p)$$

- ▶ A more symmetric version:

$$U_p(\mathbf{x}_{-p}^*; \mathbf{x}_p^*) \geq U_p(\mathbf{x}_{-p}^*; \mathbf{y}_p)$$

Why the equilibrium exists

- ▶ Based on Kakutani's fixed point theorem
- ▶ Generalizes the following: take two pieces of paper (which are 3D objects) and imagine placing them on top of each other, so that the “points” align. Crumple the top sheet without tearing it, and place it on top of the other sheet. Then, there is *always* a point that has not changed position on the two sheets.

Questions about finding Nash equilibria

- ▶ Proof of existence is non-constructive
- ▶ So how do we find it?
 - ▶ Can we find it?
 - ▶ Can we find it *efficiently*?

Do we really care?

- ▶ Obviously an “interesting” problem, but is there a significant benefit in being able to compute it?
- ▶ Arguably, equilibrium is not useful unless we can find it
 - ▶ Consider the case of a market
 - ▶ *“If your laptop can't find it [the equilibrium], then neither can the market”* [Papadimitriou]

Importance of the problem

- ▶ *“Together with factoring, the complexity of finding a Nash equilibrium is in my opinion the most important concrete open question on the boundary of P today” [Papadimitriou]*

Complexity of the problem

- ▶ Need to define a special class, PPAD, for this problem
- ▶ Turns out that finding the Nash-equilibrium is PPAD-complete
- ▶ What about NP?
 - ▶ Probably not NP-complete
 - ▶ The decision version is in P
 - ▶ Why?

Complexity of the problem

- ▶ Need to define a special class, PPAD, for this problem
- ▶ Turns out that finding the Nash-equilibrium is PPAD-complete
- ▶ What about NP?
 - ▶ Probably not NP-complete
 - ▶ The decision version is in P
 - ▶ Why?
 - ▶ Because the equilibrium *always* exists!

Polynomial parity argument

- ▶ Simple fact: Every graph has an *even* number of *odd*-degree nodes
- ▶ **Proof:** Let $W = \{v \in V : v \text{ has odd degree}\}$

$$\begin{aligned} 2|E| &= \sum_{v \in W} \deg(v) + \sum_{v \notin W} \deg(v) \\ &= \sum_{v \in W} \text{odd} + \text{even} \end{aligned}$$

- ▶ **Corollary:** If a graph has maximum degree 2, then it must have an even number of leaves

The class TFNP

- ▶ PPAD is based on the class TFNP
- ▶ Suppose we have a set of binary predicates $P(x, y)$ where

$$(\forall x)(\exists y) : P(x, y) = \text{TRUE}$$

- ▶ Problems in TFNP: Given an x , find a y so that $P(x, y)$ is TRUE
- ▶ Subclasses defined based on how we decide $(\exists y) : P(x, y)$ is TRUE

PPAD in terms of TFNP

- ▶ PPAD is defined by the following problem
 - ▶ Suppose we have an exponential-size graph where the in-degree and out-degree of each node is at most 1
 - ▶ Given any node v , we have a polynomial-time algorithm that finds the neighbours of v
 - ▶ **Problem:** Given a leaf node, output another leaf node
- ▶ Existence is guaranteed by the parity argument

Completeness of finding Nash equilibrium

- ▶ Finding a Nash equilibrium is PPAD-complete
 - ▶ For 4-player games... [Papadimitriou]

Completeness of finding Nash equilibrium

- ▶ Finding a Nash equilibrium is PPAD-complete
 - ▶ For 4-player games... [Papadimitriou]
 - ▶ ...and 3-player games [Papadimitriou]

Completeness of finding Nash equilibrium

- ▶ Finding a Nash equilibrium is PPAD-complete
 - ▶ For 4-player games... [Papadimitriou]
 - ▶ ...and 3-player games [Papadimitriou]
 - ▶ ...and even for 2-player games! [Chen]

Completeness of finding Nash equilibrium

- ▶ Finding a Nash equilibrium is PPAD-complete
 - ▶ For 4-player games... [Papadimitriou]
 - ▶ ...and 3-player games [Papadimitriou]
 - ▶ ...and even for 2-player games! [Chen]
- ▶ So finding the Nash equilibrium even for 2-player games is no easier than doing it for n -players!
- ▶ At the moment, however, not much known about how “hard” a class PPAD is

Approaches to finding equilibria

- ▶ No P algorithms known!
- ▶ Most approaches are based on solving non-linear programs (for general n)
- ▶ Completeness result means even 2-player games are not (yet) “easy” to solve
- ▶ One of the earliest algorithms for finding equilibria in 2-player games: Lemke-Howson algorithm

Lemke-Howson algorithm

- ▶ An algorithm for finding the Nash equilibrium for a game with 2 players
- ▶ Developed in 1964
 - ▶ Independent proof of why equilibrium must exist
- ▶ Worst-case exponential time, but in practise quite good performance
- ▶ Still viewed favourably, although it has been beaten of late

How we proceed

- ▶ We need to redefine a Nash equilibrium for 2-players
- ▶ Try and make a graph that lets us find equilibria easily
 - ▶ Exploiting the convenience of the alternate definition

Utility for 2-players

- ▶ Suppose that for a 2-player game, we have the mixed strategies $\mathbf{x} = (\mathbf{s}, \mathbf{t})$
- ▶ Label the strategies by $I = \{1, \dots, m\}$ for player 1, and $J = \{m + 1, \dots, m + n\}$ for player 2
- ▶ Expected utility for player p must be

$$\begin{aligned}
 U_p(\mathbf{x}) &= \sum_i \sum_j \Pr[\text{player 1 chooses } i] \times \Pr[\text{player 2 chooses } j] \\
 &\quad \times \text{Payoff for player } p \text{ when 1 plays } i \text{ and 2 plays } j \\
 &= \sum_i \sum_j \mathbf{s}(i)\mathbf{t}(j)u_p(i, j)
 \end{aligned}$$

Nash equilibria for two players

- We call $\mathbf{x}^* = (\mathbf{s}^*, \mathbf{t}^*)$ a Nash equilibrium iff

$$(\forall \mathbf{s}) \sum_i \sum_j \mathbf{s}^*(i) \mathbf{t}^*(j) u_1(i, j) \geq \sum_i \sum_j \mathbf{s}(i) \mathbf{t}^*(j) u_1(i, j)$$

$$(\forall \mathbf{t}) \sum_i \sum_j \mathbf{s}^*(i) \mathbf{t}^*(j) u_2(i, j) \geq \sum_i \sum_j \mathbf{s}^*(i) \mathbf{t}(j) u_2(i, j)$$

Claim

- ▶ **Claim:** If in a Nash equilibrium player p can play strategy i (non-zero probability), then if player p chooses to play strategy i , it is the best possible strategy
- ▶ Mathematically,

$$\mathbf{s}^*(i) > 0 \implies (\forall i_0) \sum_j \mathbf{t}^*(j) u_1(i, j) \geq \sum_j \mathbf{t}^*(j) u_1(i_0, j) \quad (1)$$

$$\mathbf{t}^*(j) > 0 \implies (\forall j_0) \sum_i \mathbf{s}^*(i) u_2(i, j) \geq \sum_i \mathbf{s}^*(i) u_2(i, j_0) \quad (2)$$

Lemma

- ▶ **Lemma:** Let $\pi_{p,i}$ denote the mixed strategy $(0, \dots, 1, \dots, 0)$ i.e. we deterministically choose strategy i for player p . Then, \mathbf{x} is a Nash Equilibrium iff

$$(\forall p, \pi_{p,i}) U_p(\mathbf{x}) \geq U_p(\mathbf{x}_{-p}; \pi_{p,i})$$

- ▶ **Proof:** (note that \implies direction is by definition)

$$\begin{aligned} U_p(\mathbf{x}_{-p}; \mathbf{y}_p) &= \sum_{i_p} \mathbf{y}_p(i_p) \left\{ \sum_{i_1 \dots i_n} \mathbf{x}_1(i_1) \dots \mathbf{x}_n(i_n) U_p(\mathbf{x}_{-p}; \pi_{p,i_p}) \right\} \\ &= \sum_{i_p} \mathbf{y}_p(i_p) U_p(\mathbf{x}_{-p}; \pi_{p,i_p}) \\ &\leq U_p(\mathbf{x}) \text{ since } \sum \mathbf{y}_p(i) = 1 \end{aligned}$$

Proof of claim

- ▶ Use the lemma: $U_p(\mathbf{x}^*) \geq U_p(\mathbf{x}_{-p}; \pi_{\mathbf{p},i})$

$$\begin{aligned} U_p(\mathbf{x}^*) &= \sum \mathbf{x}_p^*(i) U_p(\mathbf{x}^*) \text{ since } \sum \mathbf{x}_p^*(i) = 1 \\ &\geq \sum \mathbf{x}_p^*(i) U_p(\mathbf{x}_{-p}^*; \pi_{\mathbf{p},i}) \\ &= U_p(\mathbf{x}^*) \end{aligned}$$

- ▶ So, we have to conclude that

$$\sum \mathbf{x}_p^*(i) U_p(\mathbf{x}^*) = \sum \mathbf{x}_p^*(i) U_p(\mathbf{x}_{-p}^*; \pi_{\mathbf{p},i})$$

- ▶ Taking terms to one side,

$$\mathbf{x}_p^*(i) > 0 \implies U_p(\mathbf{x}^*) = U_p(\mathbf{x}_{-p}^*; \pi_{\mathbf{p},i})$$

Reformulation of Nash equilibrium - I

- ▶ So, \mathbf{x}^* is a Nash equilibrium iff
 - ▶ For player 1, equation 1 holds **or** $\Pr[\text{strategy } i] = 0$
 - ▶ For player 2, equation 2 holds **or** $\Pr[\text{strategy } j] = 0$

Reformulation of Nash equilibrium - II

- ▶ Define

$$S^i = \{\mathbf{s} : \mathbf{s}(i) = 0\}, S^j = \{\mathbf{s} : \sum_i \mathbf{s}(i)u_2(i,j) \geq \sum_i \mathbf{s}(i)u_2(i,j_0)\}$$

$$T^j = \{\mathbf{t} : \mathbf{t}(j) = 0\}, T^i = \{\mathbf{t} : \sum_j \mathbf{t}(j)u_1(i,j) \geq \sum_j \mathbf{t}(j)u_1(i_0,j)\}$$

- ▶ Then, \mathbf{x}^* is a Nash equilibrium iff

$$(\forall i)\mathbf{s} \in S^i \vee \mathbf{t} \in T^i$$

$$(\forall j)\mathbf{s} \in S^j \vee \mathbf{t} \in T^j$$

Labelling

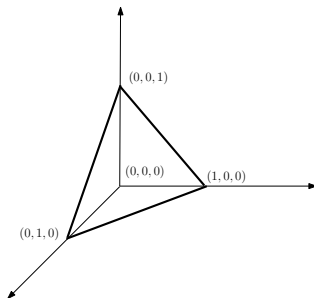
- ▶ We are claiming that $\mathbf{x} = (\mathbf{s}, \mathbf{t})$ is an equilibrium iff...
 - ▶ For any $k \in I \cup J$, either \mathbf{s} or \mathbf{t} (or maybe both) is in the appropriate region S^k or T^k
- ▶ Can think of these k 's as *labels* of points
 - ▶ $\text{Labels}(\mathbf{s}) = \{k \in I \cup J : \mathbf{s} \in S^k\}$
 - ▶ $\text{Labels}(\mathbf{t}) = \{k \in I \cup J : \mathbf{t} \in T^k\}$
 - ▶ Note: This definition also makes sense even when \mathbf{s}, \mathbf{t} aren't valid strategies

Reformulation of Nash equilibrium - III

- ▶ Natural label for $\mathbf{x} = \text{Labels}(\mathbf{s}) \cup \text{Labels}(\mathbf{t})$
- ▶ So, \mathbf{x} is a Nash equilibrium iff it is *completely labelled*

Strategy simplex

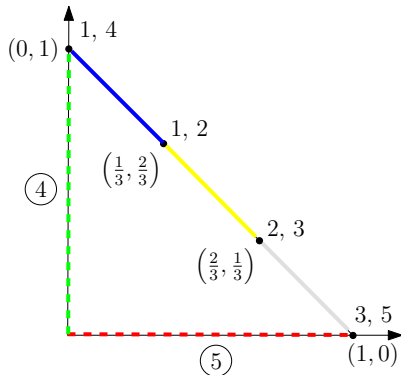
- ▶ m strategies \implies valid strategy space is an $(m - 1)$ dimensional simplex



- ▶ With labelling, we can split up the simplex into regions

Labelling - example

- ▶ For the payoff matrix $A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}$
- ▶ Label the strategy space for player 2:



Reformulation of problem

- ▶ Using the labelling definition, \mathbf{x} is a Nash equilibrium iff it is completely labelled
- ▶ \sim Point that lie in the intersection of many regions
 - ▶ But here there are two variables, and two sets of spaces
 - ▶ Can “sacrifice” one for the sake of the other
- ▶ New problem: How do we find points that are completely labelled?

High-level solution

- ▶ With labelling, we have divide the strategy spaces into sub-spaces
- ▶ We want to end up with a pair of strategies that *together* “lie in the intersection of the spaces”
- ▶ Search problem: starting off with some initial pair of strategies, we want to find the equilibrium

High-level solution

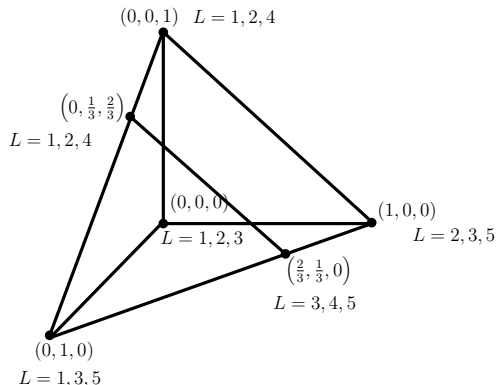
- ▶ Think of the space as a graph
- ▶ Vertices should correspond to strategy pairs
- ▶ Edges should define some well-defined change in the strategies
 - ▶ Which makes it easy to find equilibria
- ▶ Following a well-defined path on this graph should take us to equilibria
- ▶ **Problem:** How do we make such a graph?

Graph construction

- ▶ Form the graphs $G_S = (V_S, E_S)$, $G_T = (V_T, E_T)$ where:
 - ▶ $V_S \leftarrow \{\mathbf{s} \in \mathbb{R}_+^m : \sum \mathbf{s}_i \leq 1 \wedge \mathbf{s} \text{ has exactly } m \text{ labels}\}$
 - ▶ E_S between $\mathbf{s}_1, \mathbf{s}_2$ if they differ in exactly one label
 - ▶ Similarly for V_T, E_T
- ▶ Note: This is now “filling” the strategy simplex

Example

- ▶ Payoff $B = [1 \ 0; 0 \ 2; 4 \ 3]$
- ▶ Vertices lie on simplex, except for $\mathbf{0}$



Graph construction

- ▶ Form the product graph $G = G_S \times G_T$
 - ▶ $V = \{(\mathbf{s}, \mathbf{t}) : \mathbf{s} \in V_S, \mathbf{t} \in V_T\}$
 - ▶ $E = \{(\mathbf{s}_1, \mathbf{t}_1) \rightarrow (\mathbf{s}_1, \mathbf{t}_2) : \mathbf{t}_1 \rightarrow \mathbf{t}_2\} \cup \{(\mathbf{s}_1, \mathbf{t}_1) \rightarrow (\mathbf{s}_2, \mathbf{t}_1) : \mathbf{s}_1 \rightarrow \mathbf{s}_2\}$

Graph and equilibria

- ▶ So now, have a graph where vertices are strategy pairs
- ▶ Know that the equilibria is one of the vertices
- ▶ **Question:** How do we find this vertex?

Two definitions

- ▶ Define:
 - ▶ $L^-(k) :=$ vertices that have all labels except, possibly, k
 - ▶ $L :=$ vertices that have all labels
- ▶ By definition, $L \subseteq L^-(k)$
- ▶ $L = (\mathbf{0}, \mathbf{0}) \cup \{\text{Equilibria}\}$
- ▶ We can prove some properties about these sets...

Fact 1

- ▶ **Fact:** For any k , every member of L is adjacent to *exactly one* member of $L^-(k) - L$
 - ▶ For any label, every (pseudo) equilibrium is adjacent to exactly one strategy pair that is missing that label

Fact 1

- ▶ **Fact:** For any k , every member of L is adjacent to *exactly one* member of $L^-(k) - L$
 - ▶ For any label, every (pseudo) equilibrium is adjacent to exactly one strategy pair that is missing that label
- ▶ **Proof:**
 - ▶ Let $(\mathbf{s}, \mathbf{t}) \in L$. Then the label k must apply to either \mathbf{s} or \mathbf{t} , by definition
 - ▶ Suppose that \mathbf{s} is labelled with k . Then, there must be an edge between (\mathbf{s}, \mathbf{t}) and the point $(\mathbf{s}', \mathbf{t})$ where \mathbf{s}' is missing the label k
 - ▶ There is only one such \mathbf{s}' that is missing the label k - hence the neighbour is unique
 - ▶ Similar argument if \mathbf{t} is labelled with k

Fact 2

- ▶ **Fact:** For any k , every member of $L^-(k) - L$ is adjacent to *exactly two* members of $L^-(k)$
 - ▶ Every strategy pair missing exactly one label is adjacent to exactly two other strategy pairs that are potentially missing the same label

Fact 2

- ▶ **Fact:** For any k , every member of $L^-(k) - L$ is adjacent to *exactly two* members of $L^-(k)$
 - ▶ Every strategy pair missing exactly one label is adjacent to exactly two other strategy pairs that are potentially missing the same label
- ▶ **Proof:**
 - ▶ Since $|\text{Labels}(\mathbf{s}, \mathbf{t})| = m + n - 1 \neq |\text{Labels}(\mathbf{s})| + |\text{Labels}(\mathbf{t})| = m + n$, there must be a duplicate label, ℓ
 - ▶ In the graph G_S , we must have an edge from \mathbf{s} to some other point \mathbf{s}' , where \mathbf{s}' does not have the label ℓ
 - ▶ Then, the edge $(\mathbf{s}, \mathbf{t}) \rightarrow (\mathbf{s}', \mathbf{t})$ must belong to E
 - ▶ Similarly for G_T - this means that the graph G has exactly two edges that change the labelling

Putting the facts together

- ▶ $L^-(k)$ describes a subgraph of G containing (disjoint) paths and loops of G
- ▶ The endpoints of a path in G are (pseudo) equilibria
- ▶ **Problem:** How do we find this set quickly?
 - ▶ Look at this later...

Finding equilibria

- ▶ Start off at the “artificial equilibrium” $(\mathbf{0}, \mathbf{0})$
- ▶ Choose an arbitrary label $\ell \in I \cup J$
- ▶ Follow the path generated by the set $L^-(\ell)$
- ▶ When we reach the end of the path, we necessarily have stopped at an equilibria

Example

- ▶ Payoff matrices

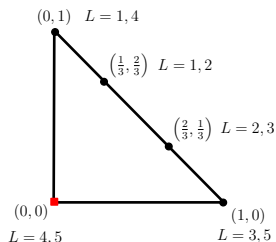
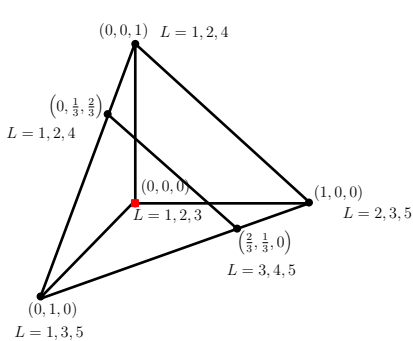
$$A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix}$$

- ▶ Choose the label 2 to be dropped i.e. move along $L^-(2)$

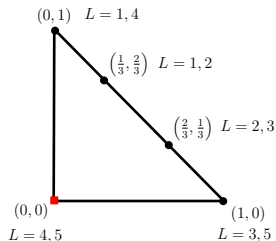
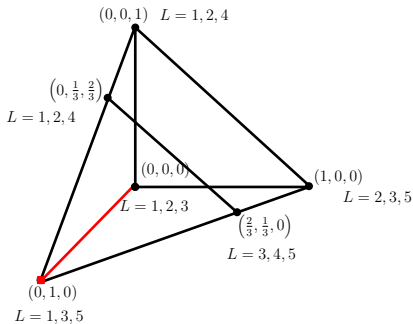
Example

- Start off at the artificial equilibrium, $((0, 0, 0), (0, 0)) \rightarrow$ labels $\{1, 2, 3\}, \{4, 5\}$



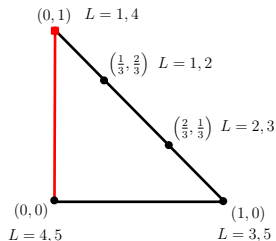
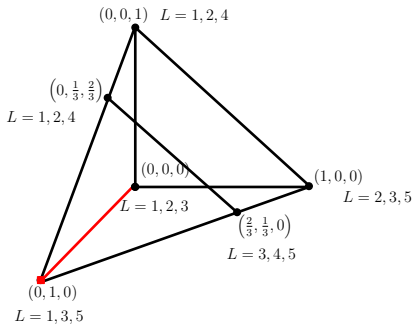
Example

- ▶ Step 1: $((0, 1, 0), (0, 0)) \rightarrow$ labels $\{1, 3, 5\}, \{4, 5\}$; duplicate is 5



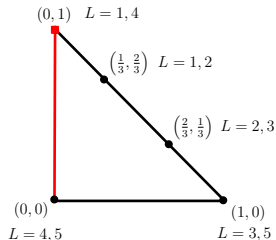
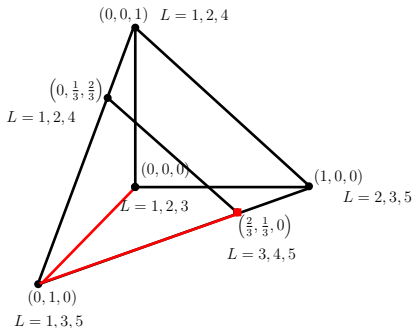
Example

- Step 2: $((0, 1, 0), (0, 1)) \rightarrow$ labels $\{1, 3, 5\}, \{1, 4\}$; duplicate is 1



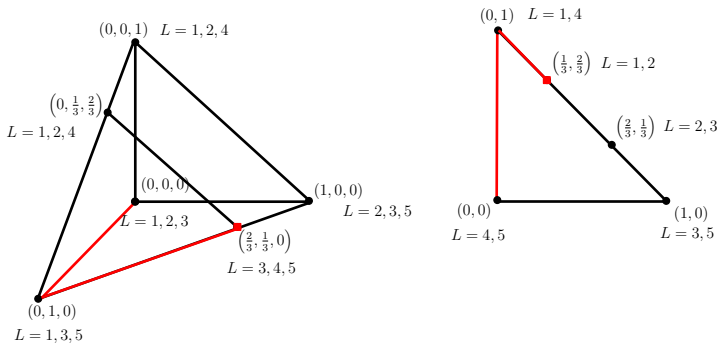
Example

- Step 3: $((\frac{2}{3}, \frac{1}{3}, 0), (0, 1)) \rightarrow$ labels $\{3, 4, 5\}, \{1, 4\}$; duplicate is 4



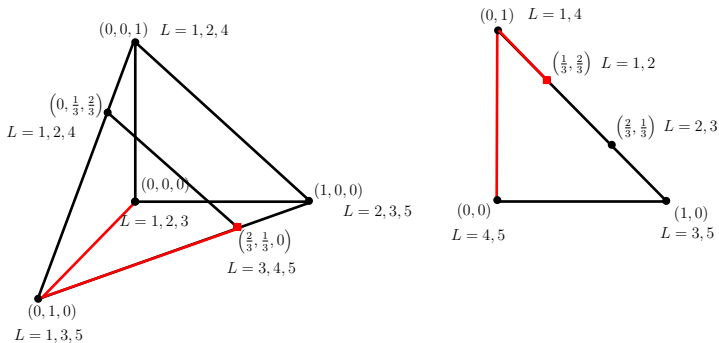
Example

- Step 4: $((\frac{2}{3}, \frac{1}{3}, 0), (\frac{1}{3}, \frac{2}{3})) \rightarrow$ labels $\{3, 4, 5\}, \{1, 2\}$



Example

- ▶ Step 4: $((\frac{2}{3}, \frac{1}{3}, 0), (\frac{1}{3}, \frac{2}{3})) \rightarrow$ labels $\{3, 4, 5\}, \{1, 2\}$
 - ▶ Completely labelled, and so an equilibria



Algorithm summary

- ▶ Consider strategies s_k that have all labels except, possibly, some label k
 - ▶ Clearly, every equilibrium belongs to this set
 - ▶ So too does the “artificial equilibrium”, $\mathbf{0}$
- ▶ Construct a graph from all strategies
- ▶ Then, one can show:
 - ▶ Each strategy missing a label is adjacent to exactly two such strategies
 - ▶ Each equilibrium is adjacent to only one strategy s_k
- ▶ It follows that:
 - ▶ Equilibria are endpoints of paths on the graph

Generating $L^-(\ell)$

- ▶ Second problem...
 - ▶ How do we find adjacent strategies?

Generating $L^-(\ell)$

- ▶ “Pivoting” approach
- ▶ Able to implicitly generate the graph, on-the-fly

Optimization problem

- ▶ Suppose we know player 2's strategy to be \mathbf{t}
- ▶ Suppose we have payoff matrices A, B
- ▶ We need to choose our strategy to maximize the payoff, i.e.

$$\begin{aligned}U_1(\mathbf{x}) &= \sum_i \sum_j \mathbf{s}(i)\mathbf{t}(j)A(i,j) \\ &= \sum_i \mathbf{s}(i)A(i, :)\mathbf{t} \\ &= \mathbf{s}^T(A\mathbf{t})\end{aligned}$$

Optimization problem

- ▶ Primal:

$$\text{maximize } \mathbf{s}^T (A\mathbf{t}) : \sum s(i) = 1, \mathbf{s} \geq \mathbf{0}$$

- ▶ Dual:

$$\text{minimize } u : u\mathbf{1} \geq A\mathbf{t}$$

Linear system

- ▶ We need to solve

$$\mathbf{s} \cdot \mathbf{1} = 1$$

$$\mathbf{t} \cdot \mathbf{1} = 1$$

$$u\mathbf{1} - A\mathbf{t} \geq \mathbf{0}$$

$$v\mathbf{1} - B^T\mathbf{s} \geq \mathbf{0}$$

$$\mathbf{s}, \mathbf{t} \geq \mathbf{0}$$

- ▶ Equilibrium iff the above, and

$$\mathbf{s}^T(u\mathbf{1} - A\mathbf{t}) = 0$$

$$\mathbf{t}^T(v\mathbf{1} - B^T\mathbf{s}) = 0$$

Linear system with slackness

- ▶ Introduce slack variables \mathbf{z}, \mathbf{w} :

$$\mathbf{s} \cdot \mathbf{1} = 1$$

$$\mathbf{t} \cdot \mathbf{1} = 1$$

$$-u\mathbf{1} + A\mathbf{t} + \mathbf{w} \cdot \mathbf{1} = \mathbf{0}$$

$$-v\mathbf{1} + B^T \mathbf{s} + \mathbf{z} \cdot \mathbf{1} = \mathbf{0}$$

$$\mathbf{s}, \mathbf{z} \geq \mathbf{0}$$

$$\mathbf{t}, \mathbf{w} \geq \mathbf{0}$$

System in matrix form

- ▶ Can write system as a matrix equation
- ▶ Consider player 1's equations - same as:

$$\mathbf{s}(1) + \dots + \mathbf{s}(m) = 1$$

$$\mathbf{b}_i^T \cdot \mathbf{s} - v + \mathbf{z}(i) = 0, i = 1, \dots, n$$

$$\mathbf{s}, \mathbf{z} \geq \mathbf{0}$$

System in matrix form

- Can write as $D\mathbf{r} = \mathbf{c}$, where

$$D = \begin{bmatrix} 1 & 1 & \dots & 1 & 0 & 0 & \dots & 0 \\ b_{11} & b_{21} & \dots & b_{m1} & -1 & 1 & \dots & 0 \\ & & & \vdots & & & & \\ b_{1n} & b_{2n} & \dots & b_{mn} & -1 & 0 & \dots & 1 \end{bmatrix}$$

$$\mathbf{r} = \begin{bmatrix} \mathbf{s}(1) \\ \vdots \\ \mathbf{s}(m) \\ v \\ \mathbf{z}(1) \\ \vdots \\ \mathbf{z}(n) \end{bmatrix}, \text{ and } \mathbf{c} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Connection with labelling

- ▶ Let the basis of the column space of D be $\{D_\beta\}$
- ▶ Call r_j ...
 - ▶ A *basic variable* if D_j is in the basis
 - ▶ A *non-basic variable* otherwise; let $r_j = 0$ in this case
- ▶ Labels correspond to non-basic variables
 - ▶ $r_1, \dots, r_m = 0 \implies \mathbf{s}(i) = 0$ i.e. strategy is not played
 - ▶ $r_{m+2}, \dots, r_{m+n+1} = 0 \implies \mathbf{z}(i) = 0$ i.e. strategy is best-response
- ▶ So, removing a label corresponds to making a variable strictly positive

Rewriting system

- ▶ System is the same as

$$D_{\beta}^{-1}(D\mathbf{r}) = D_{\beta}^{-1}\mathbf{c}$$

- ▶ **Fact:** $D_{\beta}^{-1}D = I + \sum_{j \notin \beta} D_{\beta}^{-1}D_j$
- ▶ System is the same as

$$\mathbf{r}_{\beta} = D_{\beta}^{-1}\mathbf{c} - \sum_{j \notin \beta} D_{\beta}^{-1}D_j r_j$$

Why rewrite?

- ▶ What is wrong with the standard representation of the system?
- ▶ Nothing “wrong” with it...
 - ▶ Just that the new system makes it easier to find strategies with different labels
 - ▶ i.e. Generating the set $L^-(\ell)$
 - ▶ This is the “pivot” operation

Example

- ▶ Use earlier payoff matrix
- ▶ Matrix D for player 1 is

$$D = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 4 & -1 & 1 & 0 \\ 0 & 2 & 3 & -1 & 0 & 1 \end{bmatrix}$$

- ▶ Choose column-space $\{2, 4, 5\}$
 - ▶ Setting $x_1, x_3 = 0$
 - ▶ Setting $z_2 = 0$

$$D_\beta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 2 & -1 & 0 \end{bmatrix}$$

Example

- ▶ Solution:

$$r_\beta = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix} - \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} r_1 - \begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix} r_3 - \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix} r_6$$

- ▶ Or equivalently:

$$x_2 = 1 - x_1 - x_3$$

$$v = 2 - 2x_1 + x_3 + z_2$$

$$z_1 = 2 - 3x_1 - 3x_3 + z_2$$

Example

- ▶ Can read off solution from here
 - ▶ $\mathbf{s} = (0, 1, 0)$
 - ▶ $v = 2$
 - ▶ $z_1 = 2, z_2 = 0$
- ▶ Can read the labels: 1, 3, 5
- ▶ Do the same thing for player 2
- ▶ **Result:** We get a pair of strategies (\mathbf{s}, \mathbf{t}) , and we know what their labels are
- ▶ **Question:** How do we find the “adjacent” strategy, as per the graph?

Finding adjacent strategy

- ▶ Traversal step in the algorithm was...
 - ▶ Find the duplicate label in the two strategies
 - ▶ Drop the label from player 1 and player 2 alternately
- ▶ Not hard to find duplicate (just read off from solution)
- ▶ How to drop label?

Finding adjacent strategy

- ▶ Traversal step in the algorithm was...
 - ▶ Find the duplicate label in the two strategies
 - ▶ Drop the label from player 1 and player 2 alternately
- ▶ Not hard to find duplicate (just read off from solution)
- ▶ How to drop label?
 - ▶ Just change the appropriate variable to be positive

Finding adjacent strategy

- ▶ Traversal step in the algorithm was...
 - ▶ Find the duplicate label in the two strategies
 - ▶ Drop the label from player 1 and player 2 alternately
- ▶ Not hard to find duplicate (just read off from solution)
- ▶ How to drop label?
 - ▶ Just change the appropriate variable to be positive
 - ▶ As large as possible while satisfying constraints

Example of pivoting

- ▶ Starting equation for player 1:

$$x_2 = 1 - x_1 - x_3$$

$$v = 2 - 2x_1 + x_3 + z_2$$

$$z_1 = 2 - 3x_1 - 3x_3 + z_2$$

- ▶ Starting equation for player 2:

$$y_2 = 1 - y_1$$

$$u = 6 - 6y_1 + w_1$$

$$w_2 = 1 - 3y_1 + w_1$$

$$w_3 = 3 - 6y_1 + w_1$$

Example of pivoting

- ▶ Solve to get $((0, 1, 0), (0, 1))$
 - ▶ $\mathbf{s} = (0, 1, 0), \mathbf{z} = (2, 0)$
 - ▶ $\mathbf{w} = (0, 1, 3), \mathbf{t} = (0, 1)$
- ▶ $x_1 = 0$ and $w_1 = 0$, so label 1 is duplicate
 - ▶ So we look to get rid of it
 - ▶ \implies Modify x_1 while satisfying non-negativity constraints
- ▶ Relevant equations:

$$x_2 = 1 - x_1$$

$$z_1 = 2 - 3x_1$$

Example of pivoting

- ▶ For first equation: $x_1 \in (0, 1]$
- ▶ For second equation: $x_1 \in (0, \frac{2}{3}]$
- ▶ Clearly, $x_1 \rightarrow \frac{2}{3}$
 - ▶ $x_2 \rightarrow \frac{1}{3}$
 - ▶ But, $z_1 \rightarrow 0!$
 - ▶ As a result, we see that label 4 is duplicate
- ▶ Another duplicate \implies we repeat the process!

Updating system

- ▶ Now x_1 should appear on the LHS, since it is no longer zero
- ▶ New column basis is $\{D_1, D_2, D_4\}$

$$D_\beta = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & 2 & -1 \end{bmatrix}$$

Updating system

- ▶ New system is

$$v = \frac{2}{3} - 3x_3 - \frac{2}{3}z_1 + \frac{1}{3}z_2$$

$$x_1 = \frac{1}{3} - x_3 - \frac{1}{3}z_1 + \frac{1}{3}z_2$$

$$x_2 = \frac{1}{3} + \frac{1}{3}z_1 - \frac{1}{3}z_2$$

Complementary pivoting

- ▶ Newly added label will be duplicate
- ▶ e.g. Label 4 is added, and is a duplicate
- ▶ Make y_1 positive
 - ▶ $y_2 = 1 - y_1 \implies y_1 \in (0, 1]$
 - ▶ $w_2 = 1 - 3y_1 + w_1 \implies y_1 \in (0, \frac{1}{3}]$
- ▶ So, new strategy is $(\frac{1}{3}, \frac{2}{3})$
 - ▶ Now completely labelled
 - ▶ Hence, we have found the equilibria

Performance of Lemke-Howson





- ▶ Worst-case exponential running time
- ▶ In practise, reasonably fast
 - ▶ Of late, been beaten by a few competing algorithms e.g. [SHNP]

Problems?

- ▶ Does not generalize to $n > 2$ players
- ▶ Sometimes, equilibria may be out of reach

Other approaches

- ▶ *Many* more techniques...
 - ▶ Local-search techniques [SHNP]
 - ▶ Mixed integer programming
 - ▶ Global Newton optimization
 - ▶ Computer algebra
 - ▶ Markov Random Fields
 - ▶ etc...
- ▶ Quite a few generalize to more than 2 players
- ▶ Nothing (as yet) tells us about the boundary of P!

-  Xi Chen, Xiaotie Deng, and Shang hua Teng.
Settling the complexity of computing two-player nash equilibria.
Technical report, 2007.
-  Ryan Porter, Eugene Nudelman, and Yoav Shoham.
Simple search methods for finding a nash equilibrium.
In *AAAI*, pages 664–669, 2004.
-  R. Savani and B. von Stengel.
Exponentially many steps for finding a nash equilibrium in a bimatrix game, 2004.
-  Lloyd S. Shapely.
A note on the Lemke-Howson algorithm.
Mathematical Programming Study 1:Pivoting and Extensions,
pages 175 – 189, 1974.



B. von Stengel.

Computing equilibria for two-person games.

Handbook of Game Theory, 3, 2002.