Nash equilibria: Complexity and Computation

INFO4011 Algorithmic Game Theory

Aditya Menon

August 7, 2007

Introduction / Recap
- Nash equilibrium
- A formal definition

Complexity
- Characterizing complexity
- PPAD
- Completeness

Lemke-Howson algorithm
- Useful claim
- Reformulations of equilibria
- Labelling
- Graph construction
- Important facts
- An example run

Approximate equilibria
- Constant epsilon methods
- Arbitrary epsilon methods

Summary

References

Nash-equilibrium

- 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

Nash-equilibrium

- 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 $\mathbf{x_p} = (x_{p1}, \ldots, x_{pn_p})$
  - The ith entry gives the probability that player p plays move i
- Let $\mathbf{x} := (\mathbf{x_1}, \ldots, \mathbf{x_n})$ 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 $\mathbf{x}$:

$$U_p(\mathbf{x}) = \sum_s \mathbf{x_1}(s_1) \ldots \mathbf{x_n}(s_n) u_p(s_1, \ldots, s_n)$$

  - $u_p(s_1, \ldots, s_n)$ is the (deterministic) utility for player p when player q plays s_q

Formal definition

- We say that $\mathbf{x}^* = (\mathbf{x_1}^*, \ldots, \mathbf{x_n}^*)$ 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:

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

Questions about finding Nash equilibria

- Proof of existence was via a fixed point theorem
  - Non-constructive
- So how do we find it?
  - And can we find it *efficiently*?

Complexity of the problem

- NASH does not fall into a standard complexity class
- 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?

What about NP?

- Probably not NP-complete
- The decision version is in P
  - Why?
  - Because the equilibrium *always* exists!

The class TFNP

- Suppose we have a set of polynomial-time computable binary predicates $P(x, y)$ where

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

- Problems in TFNP: Given an x, find a y so that $P(x, y)$ is TRUE
  - Can be thought of as "NP search problems where a solution is guaranteed"
- 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 (complete) problem:

Problem

Suppose we have an exponential-size directed graph $G = (V, E)$, where the in-degree and out-degree of each node is at most 1. Given any node $v \in V$, suppose we have a polynomial-time algorithm that finds the neighbours of v. Now suppose we are given a leaf node w - output another leaf node $w' \neq w$.

- Existence of another leaf node is guaranteed by the *parity argument*
  - Hence the name

Polynomial parity argument

Theorem

Every graph has an even number of odd-degree nodes

Polynomial parity argument

Theorem

Every graph has an even number of odd-degree nodes

- **Proof**: Let $W = \{v \in V : v \text{ has odd degree }\}$

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

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

PPAD-completeness

- Some other PPAD complete problems are...
  - Finding a Sperner triangle
  - Finding a Brouwer fixed point
  - And finding a Nash equilibrium!

Completeness of finding Nash equilibrium

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

Completeness of finding Nash equilibrium

- Finding a Nash equilibrium is PPAD-complete
  - For 4-player games... [3]
  - ...and 3-player games... [1, 7]

Completeness of finding Nash equilibrium

- Finding a Nash equilibrium is PPAD-complete
  - For 4-player games... [3]
  - ...and 3-player games... [1, 7]
  - ...and even for 2-player games! [2]

Completeness of finding Nash equilibrium

- Finding a Nash equilibrium is PPAD-complete
  - For 4-player games... [3]
  - ...and 3-player games... [1, 7]
  - ...and even for 2-player games! [2]
- 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
  - i.e. Where does it lie w.r.t. P?

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 [16, 14]
- Developed in 1964
  - Independent proof of why equilibrium must exist
- Worst-case exponential time [13], but in practise quite good performance

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 $I = \{1, \ldots, m\}$ for player 1, and $J = \{m+1, \ldots, m+n\}$ for player 2
- Expected utility for player p must be

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

Nash equilibria for two players

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

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

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

A useful claim

Claim

If in a Nash equilibrium player p *can* play strategy i (non-zero probability), then strategy i is a best-response strategy

A useful claim

Claim

If in a Nash equilibrium player *p* *can* play strategy *i* (non-zero probability), then strategy *i* is a best-response strategy

- Mathematically,

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

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

Proof?

- We need a lemma to prove this
- We show that it is sufficient that we simply beat *pure* strategies of other players

Lemma

Lemma

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

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

Lemma

Lemma

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

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

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

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

Proof of claim

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

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

- So, we deduce that

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

- Taking terms to one side,

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

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

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

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

- 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

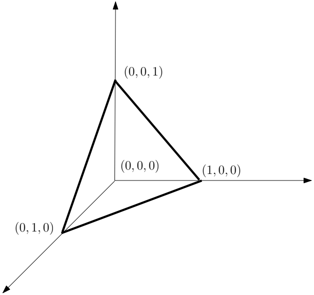
- We are claiming that $\mathbf{x} = (\mathbf{s}, \mathbf{t})$ is an equilibrium iff...
  - For any $k \in I \cup J$, either $\mathbf{s}$ or $\mathbf{t}$ (or maybe both) is in the appropriate region S^k or T^k
- Can think of these k's as *labels* of strategies
  - $\text{Labels}(\mathbf{s}) = \{k \in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

Reformulation of Nash equilibrium - III

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

Strategy simplex

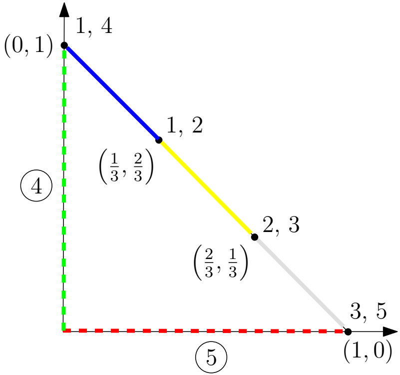
- m strategies $\implies$ valid strategy space is an $(m-1)$ dimensional simplex



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

Labelling - example

- For the payoff matrix $A = [0 \quad 6; 2 \quad 5; 3 \quad 3]$
- 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
- **New problem**: How do we find points that are completely labelled?

High-level solution

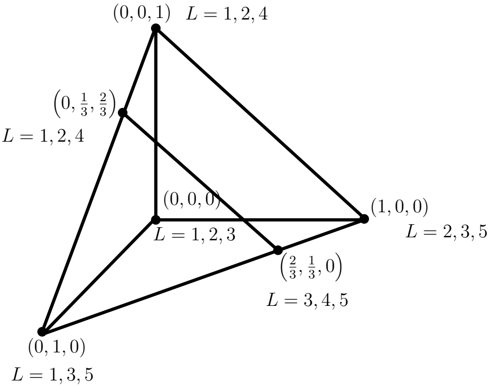
- Think of the space as a graph
  - Vertices should correspond to strategy pairs
  - Edges correspond to some change in the strategies
- We want to move from some starting pair to an equilibrium
- So, we need to carefully choose edges
  - Edges should define some special change in the strategies
  - Should make it easy to find equilibria
- **Problem**: How do we make such a graph?
  - What is a good rule for making edges?

Graph construction

- Form the graphs $G_S = (V_S, E_S)$, $G_T = (V_T, E_T)$ where:
  - $V_S \leftarrow \{\mathbf{s} \in \mathbb{R}^m_+ : \mathbf{s}$ is inside the simplex, and $\mathbf{s}$ has exactly m labels $\}$
  - Edge 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 $B = [1 \quad 0; 0 \quad 2; 4 \quad 3]$



Why zero?

- **Fact**: Vertices lie on simplex, except for $\mathbf{0}$
- Zero is the only "non-strategy" vertex we select
- Why didn't we just specify that $\sum \mathbf{s}_i = 1$?
  - The value of zero will be revealed later!
  - For now, notice that $(\mathbf{0}, \mathbf{0})$ is completely labelled, but is not an equilibrium...

Graph construction

- Form the product graph $G = G_S \times G_T$
  - $V = \{(\mathbf{s}, \mathbf{t}) : \mathbf{s} \in V_S, \mathbf{t} \in V_T\}$
  - $E = \{(\mathbf{s_1}, \mathbf{t_1}) \to (\mathbf{s_1}, \mathbf{t_2}) : \mathbf{t_1} \to \mathbf{t_2}\} \cup \{(\mathbf{s_1}, \mathbf{t_1}) \to (\mathbf{s_2}, \mathbf{t_1}) : \mathbf{s_1} \to \mathbf{s_2}\}$
- Now we have vertices corresponding to pairs of mixed strategies

Graph motivation

- We know that the equilibria are completely labelled
- We know that G must therefore contain the equilibria as vertices
- We know that edges between vertices only modify one label
- **Question**: Can we traverse the graph so that we find an equilibria?

Two important sets

- Define:
  - $L^{-}(k) :=$ vertices that have all labels except, possibly, k
  - $L :=$ vertices that have all labels
- By definition, $L \subseteq L^{-}(k)$
- $L = (\mathbf{0}, \mathbf{0}) \cup \{\text{Equilibria}\}$
  - So we call $(\mathbf{0}, \mathbf{0})$ the "pseudo" equilibrium
- 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 $L^{-}(k) - L$. That is, 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 $L^{-}(k) - L$. That is, 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 **s** or **t**, by definition
  - Suppose that **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 **t** is labelled with k

Fact 2

Fact

For any k, every member of $L^{-}(k) - L$ is adjacent to *exactly two* members of $L^{-}(k)$. That is, 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 $L^-(k)$. That is, 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?
  - Touch on this later

The value of zero

- We know that if we start at a (pseudo) equilibrium, we will end up at a different (pseudo) equilibrium
- Now we are glad we added the pseudo equilibrium $(\mathbf{0}, \mathbf{0})$
  - It gives us a constant, convenient starting point!
  - Otherwise, only if we *already* knew an equilibrium could we find another

Finding equilibria

- Start off at the pseudo-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 will necessarily have stopped at an equilibrium

Example

- Payoff matrices (from [16])

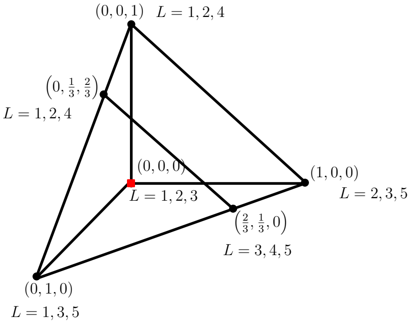
$$A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}$$

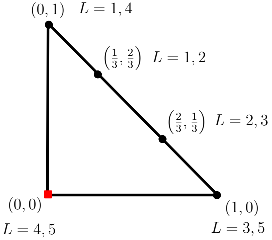
$$B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix}$$

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

Example

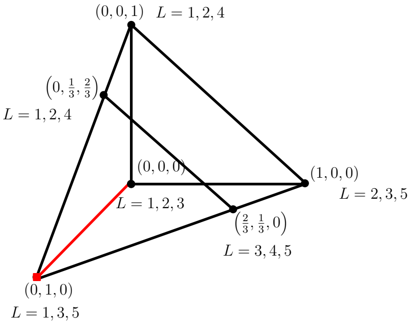
- Start off at the artificial equilibrium, $((0,0,0),(0,0)) \longrightarrow$ labels $\{1,2,3\},\{4,5\}$

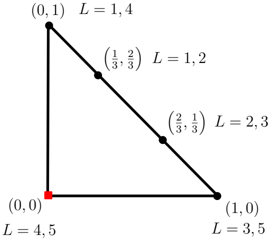




Example

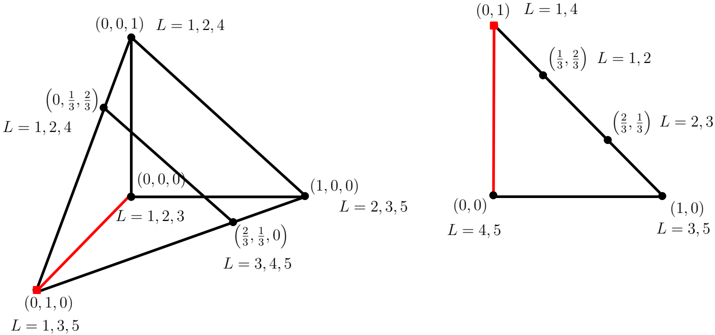
- Step 1: $((0,1,0),(0,0)) \longrightarrow$ labels $\{1,3,5\},\{4,5\}$; duplicate is 5





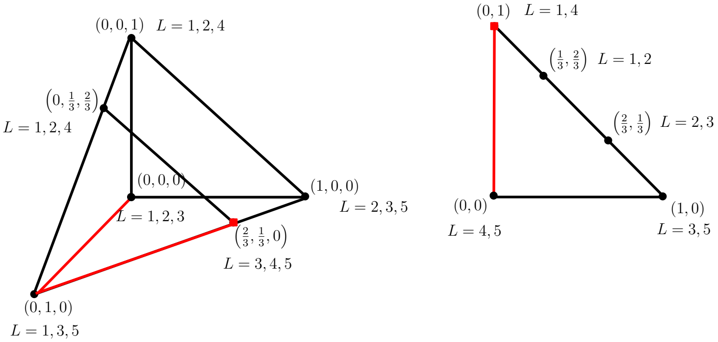
Example

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



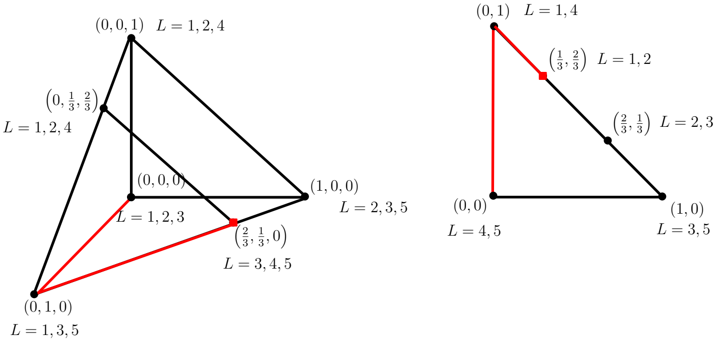
Example

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



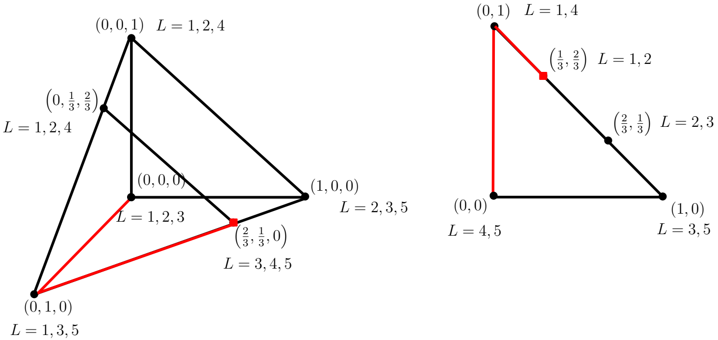
Example

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



Example

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



Algorithm summary

- Consider strategies in $L^{-}(\ell)$ that have all labels except, possibly, some label ℓ
  - Clearly, every equilibrium belongs to this set
  - So too does the pseudo equilibrium, $(\mathbf{0}, \mathbf{0})$
- Construct a graph from all such 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
- It follows that:
  - Equilibria are endpoints of paths along $L^{-}(\ell)$ on the graph

Generating $L^{-}(\ell)$

- Second problem...
  - How do we find adjacent strategies?
- "Pivoting" approach
  - Write problem as a matrix equation
  - Labels correspond to zero entries in solution vector
- Able to implicitly generate the graph, on-the-fly
- Details in [16]!

Performance of Lemke-Howson

- Worst-case exponential running time
  - In practise, reasonably fast
  - c.f. Simplex algorithm
- Does not generalize to $n > 2$ players
- Sometimes, equilibria may be out of reach

Other approaches

- Many more techniques, of diverse types...
  - Local-search techniques [11]
  - Mixed integer programming [12]
  - Computer algebra [8]
  - Markov Random Fields [6]
  - etc...
- Quite a few generalize to more than 2 players
- Nothing (as yet) tells us about the boundary of P!

Other avenues

- So finding a Nash equilibria is not currently easy
  - It is not known how to do it in polynomial time
- What about an *approximate* solution?
  - Hopefully, these may permit polynomial algorithms...

Approximate equilibria

- Standard definition of approximate equilibria is one of additive error
- We call $\mathbf{x}^*$ an ϵ-approximate Nash equilbria if, for every player p and for any mixed strategy $\mathbf{y_p}$, we have

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

  - We don't lose more than ϵ by changing our current strategy
- A Nash equilibrium is a "0-approximate" Nash equilibrium

A useful fact

Fact

If a game with payoff matrices R, C has a Nash equilibria $(\mathbf{s}^*, \mathbf{t}^*)$, then the game $\alpha R + \beta, \gamma C + \delta$ has the same equilibria, for any $\alpha, \gamma > 0, \beta, \delta \in \mathbb{R}$.

- This means that we can normalize any game so that the payoffs are between 0 and 1
- Makes some of the analysis simpler

Simple methods for constant ϵ

- Daskalakis [5] showed how to find a $\frac{1}{2}$-approximate equilibria
- Kontogiannis [9] gave a way to find a $\frac{3}{4}$-approximate equilibria, and then a parametrized approximate equilibria
- Both in polynomial time!

A $\frac{1}{2}$-approximate equilibria

- Say we have a two-player game, with payoff matrices R, C (row, column) for players 1 and 2
- Pick an arbitrary strategy (row) for the first player - say i
- Define

$$j := \text{argmax}_{j_0} C_{ij_0}$$

$$k := \text{argmax}_{k_0} R_{k_0 j}$$

  - j is the best-response for player 2
  - k is the best-response to the best-response for player 1

A $\frac{1}{2}$-approximate equilibria

Claim

The strategy-pair $\left(\frac{\pi_i+\pi_k}{2}, \pi_j\right)$ is a $\frac{1}{2}$-approximate Nash equilibria.

A $\frac{1}{2}$-approximate equilibria

Claim

The strategy-pair $\left(\frac{\pi_i+\pi_k}{2}, \pi_j\right)$ is a $\frac{1}{2}$-approximate Nash equilibria.

- **Proof**:
  - Row player's payoff is $\mathbf{s}^*.(R\mathbf{t}^*) = \frac{R_{ij}+R_{kj}}{2}$
  - Column player's payoff is $\mathbf{s}^*.(C\mathbf{t}^*) = \frac{C_{ij}+C_{kj}}{2}$
  - Row player's incentive to deviate is

$$R_{kj} - \frac{R_{ij}+R_{kj}}{2} \leq \frac{R_{kj}}{2} \leq \frac{1}{2}$$

  - Column player's incentive to deviate is

$$\frac{C_{ij'}+C_{kj'}}{2} - \frac{C_{ij}+C_{kj}}{2} \leq \frac{C_{kj'}-C_{kj}}{2} \leq \frac{1}{2}$$

Parameterized approximation

- Kontogiannis [9] gave a simple way to find a $\frac{3}{4}$-approximate equilibrium
- We look at how he finds a $\frac{2+\lambda}{4}$-approximate equilibrium
  - $\lambda \in [0, 1)$ (unfortunately!) not arbitrary
- **Idea**: Define a "good" pair of linear programs
  - Equilibria solve the programs, but not necessarily optimally
  - Relate optimal solution of LPs to Nash equilibria

Parameterized approximation

- Consider the linear programs

$$\begin{array}{cc} \text{minimize } p : & \text{minimize } q : \\ (\forall i)(R\mathbf{t})_i \leq p & (\forall j)(\mathbf{s}C)_j \leq q \\ \sum \mathbf{t}_j = 1 & \sum \mathbf{s}_i = 1 \\ \mathbf{t} \geq \mathbf{0} & \mathbf{s} \geq \mathbf{0} \end{array}$$

- Solutions will be

$$\mathbf{t} = \text{argmin}\,\{\max_i (R\mathbf{t})_i\}$$

$$\mathbf{s} = \text{argmin}\,\{\max_j (\mathbf{s}C)_j\}$$

Another interesting property

Theorem

Suppose $(\mathbf{s}^*, \mathbf{t}^*)$ *is a Nash equilibrium. Then,*

$$\mathbf{s}^*.(R\mathbf{t}^*) = max_i(R\mathbf{t}^*)_i$$

$$\mathbf{s}^*.(C\mathbf{t}^*) = max_j(\mathbf{s}^* C)_j$$

That is, the expected payoff for both players equals the maximal payoff under pure strategies

Proof

- Follows easily from the linearity of the sum. Firstly,

$$\sum \mathbf{s}^*(i)(R\mathbf{t}^*)_i \leq \sum \mathbf{s}^*(i)\max_i(R\mathbf{t}^*)_i = \max_i(R\mathbf{t}^*)_i$$

  Secondly, let

$$i_0 = \operatorname{argmax}_i(R\mathbf{t}^*)_i$$

  and then the Nash property tells us

$$\sum \mathbf{s}^*(i)(R\mathbf{t}^*)_i \geq \sum \pi_{\mathbf{i_0}}(i)(R\mathbf{t}^*)_i = \max_i(R\mathbf{t}^*)_i$$

Optimal solutions to LPs

- Suppose the optimal solutions are $(p_0, \mathbf{t}^*), (q_0, \mathbf{s}^*)$
- That means for some r, c, the values are attained:

$$(R\mathbf{t}^*)_r = p_0$$

$$(\mathbf{s}^* C)_c = q_0$$

- That is,

$$r = \text{argmax}_i (R\mathbf{t}^*)_i$$

$$s = \text{argmax}_j (\mathbf{s}^* C)_j$$

Equilibrium solutions to LPs

- Now consider the equilibria $(\mathbf{s_1}, \mathbf{t_1})$, $(\mathbf{s_2}, \mathbf{t_2})$, where...
  - $(\mathbf{s_1}, \mathbf{t_1})$ gives the minimal payoff λ_1 for player 1
  - $(\mathbf{s_2}, \mathbf{t_2})$ gives the minimal payoff λ_2 for player 2
- By the theorem, $(\lambda_1, \mathbf{t_1})$ and $(\lambda_2, \mathbf{s_2})$ are feasible solutions to the corresponding LPs
  - This is because e.g. $\max_i (R\mathbf{t})_i$ is the payoff for player 1
- As a consequence, $p_0, q_0 \leq \lambda = \max\{\lambda_1, \lambda_2\}$
  - Equilibrium solution is not necessarily the optimal one

The strategy

- Use the strategies $(\mathbf{s}, \mathbf{t})$, with:

$$\mathbf{s}(i) = \frac{\mathbf{s}^*(i)}{2}, i \neq r$$
$$\mathbf{s}(r) = \frac{1}{2} + \frac{\mathbf{s}^*(r)}{2}$$
$$\mathbf{t}(j) = \frac{\mathbf{t}^*(j)}{2}, j \neq c$$
$$\mathbf{t}(c) = \frac{1}{2} + \frac{\mathbf{t}^*(c)}{2}$$

- We are boosting the "optimal" strategies
- Proof of $\frac{2+\lambda}{4}$-approximation follows easily

Some other results

- Daskalakis [4] gave a $(1-\phi)+\epsilon \approx 0.38+\epsilon$ approximate equilibria algorithm
  - Solving a linear system
- Tsaknakis [15] recently gave the best known result
  - Finds a $\frac{1}{3}$-approximate equilibrium in polynomial time
  - Based on a steepest descent
  - Indicates why this might be a barrier value in terms of complexity

Arbitrary ϵ?

- What about when ϵ is not fixed?
- Lipton [10] gave a simple algorithm for finding a sparse approximate equilibria
- Based on sampling theory
- First sub-exponential algorithm known for arbitrary ϵ!
- Interesting result on the nature of approximate equilibria
  - There is always an approximate equilibrium that is sparsely populated

Sparse approximate equilibria

- Call a strategy *uniform* if the probability of all possible moves (with non-zero probability) are equal
- Call the set of possible (non-zero) strategies the *support* of a mixed strategy

Sparse approximate equilibria

- Call a strategy *uniform* if the probability of all possible moves (with non-zero probability) are equal
- Call the set of possible (non-zero) strategies the *support* of a mixed strategy

Theorem

For any Nash equilibrium $(\mathbf{s}^*, \mathbf{t}^*)$*, and any* $c \geq 12$*, there is an* ϵ*-approximate equilibrium* $(\mathbf{s}, \mathbf{t})$ *such that* $\mathbf{s}, \mathbf{t}$ *have support of size* $c\frac{\log n}{\epsilon^2}$*, where both are uniform strategies, and this new strategy pair approximates the payoffs in the equilibrium case:*

$$|\mathbf{s}.(R\mathbf{t}) - \mathbf{s}^*.(R\mathbf{t}^*)| < \epsilon$$

$$|\mathbf{s}.(C\mathbf{t}) - \mathbf{s}^*.(C\mathbf{t}^*)| < \epsilon$$

Proof idea

- Use the probabilistic method
- We want the probability that a randomly picked $\mathbf{x}, \mathbf{y}$ will satisfy

$$(|\mathbf{s}.(R\mathbf{t}) - \mathbf{s}^*.(R\mathbf{t}^*)| < \epsilon) \wedge (|\mathbf{s}.(C\mathbf{t}) - \mathbf{s}^*.(C\mathbf{t}^*)| < \epsilon) \wedge$$
$$(|\pi_{\mathbf{i}}.(R\mathbf{t}) - \mathbf{s}.(R\mathbf{t})| < \epsilon) \wedge (|\mathbf{s}.(C\pi_{\mathbf{j}}) - \mathbf{s}.(C\mathbf{t})| < \epsilon)$$

is positive, for any i, j

Proof idea

- Use the fact that e.g.

$$(|\mathbf{s}^*.(R\mathbf{t}^*) - \mathbf{s}.(R\mathbf{t}^*)| < \epsilon/2) \wedge (\mathbf{s}.(R\mathbf{t}^*) - |\mathbf{s}.(R\mathbf{t})| < \epsilon/2)$$
$$\implies |\mathbf{s}.(R\mathbf{t}) - \mathbf{s}^*.(R\mathbf{t}^*)| < \epsilon$$

- Can show e.g.

$$\mathbb{E}\left[(\mathbf{s}.(R\mathbf{t}^*))_i\right] = \mathbf{s}^*.(R\mathbf{t}^*)$$

- Use standard concentration bounds for 0-1 variables to show

$$\Pr\left[|\mathbf{s}^*.(R\mathbf{t}^*) - \mathbf{s}.(R\mathbf{t}^*)| \geq \epsilon/2\right] \leq 2e^{-k\epsilon^2/8}$$

- End up with a sum of exponentially small terms, so that

$$\Pr\left[\text{Conditions fail}\right] < 0$$

How do we use it?

- Suggests a simple algorithm for finding an ϵ-approximate, sparse equilibria
- Just enumerate all possible k-uniform strategies for some fixed $k \geq \frac{12 \log n}{\epsilon^2}$
  - Theorem guarantees that at least one of these strategies will ϵ-approximate a Nash equilibrium
  - To test the ϵ-approximation, check deviation to pure strategies
- Runtime is $\binom{n+k-1}{k}^2 = O\left(n^{2k}\right) = O\left(n^{\log n/\epsilon^2}\right)$
  - Unordered selection, with repetition, of k things from n things
  - This is a sub-exponential algorithm

Summary

- We don't know how to find Nash equilibria in P time, in general
- Even 2-player games are hard!
- We can solve 2-player games in "average" case polynomial time
  - e.g. Lemke-Howson algorithm
- Approximate-equilibria permit polynomial solutions
  - (Some) Constant-approximations are in P
  - General ϵ is sub-exponential at least
- Computing equilibria is an important problem in theoretical CS!

Xi Chen and Xiaotie Deng.
3-nash is PPAD-complete.
Electronic Colloquium on Computational Complexity (ECCC), (134), 2005.

Xi Chen, Xiaotie Deng, and Shang hua Teng.
Settling the complexity of computing two-player nash equilibria.
Technical report, 2007.

Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou.
The complexity of computing a nash equilibrium.
In *STOC '06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 71–78, New York, NY, USA, 2006. ACM Press.

Constantinos Daskalakis, Aranyak Mehta, and Christos Papadimitriou.
Progress in approximate nash equilibria.
In *EC '07: Proceedings of the 8th ACM conference on Electronic commerce*, pages 355–358, New York, NY, USA, 2007. ACM Press.

Constantinos Daskalakis, Aranyak Mehta, and Christos H. Papadimitriou.
A note on approximate nash equilibria.
In Paul G. Spirakis, Marios Mavronicolas, and Spyros C. Kontogiannis, editors, *WINE*, volume 4286 of *Lecture Notes in Computer Science*, pages 297–306. Springer, 2006.

Constantinos Daskalakis and Christos H. Papadimitriou.
Computing pure nash equilibria in graphical games via markov random fields.
In *EC '06: Proceedings of the 7th ACM conference on Electronic commerce*, pages 91–99, New York, NY, USA, 2006. ACM Press.

Konstantinos Daskalakis and Christos H. Papadimitriou.
Three-player games are hard.
Electronic Colloquium on Computational Complexity (ECCC), (139), 2005.

Ruchira S. Datta.
Using computer algebra to find nash equilibria.
In *ISSAC '03: Proceedings of the 2003 international symposium on Symbolic and algebraic computation*, pages 74–79, New York, NY, USA, 2003. ACM Press.

Spyros C. Kontogiannis, Panagiota N. Panagopoulou, and Paul G. Spirakis.
Polynomial algorithms for approximating nash equilibria of bimatrix games.
In Paul G. Spirakis, Marios Mavronicolas, and Spyros C. Kontogiannis, editors, *WINE*, volume 4286 of *Lecture Notes in Computer Science*, pages 286–296. Springer, 2006.

Richard J. Lipton, Evangelos Markakis, and Aranyak Mehta.
Playing large games using simple strategies.
In *EC '03: Proceedings of the 4th ACM conference on Electronic commerce*, pages 36–41, New York, NY, USA, 2003. ACM Press.

Ryan Porter, Eugene Nudelman, and Yoav Shoham.
Simple search methods for finding a nash equilibrium.
In *AAAI*, pages 664–669, 2004.

Tuomas Sandholm, Andrew Gilpin, and Vincent Conitzer.
Mixed-integer programming methods for finding nash equilibria.
In *AAAI*, pages 495–501, 2005.

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.

Haralampos Tsaknakis and Paul Spirakis.
Computing 1/3-approximate nash equilibria of bimatrix games in polynomial time.
Electronic Colloquium on Computational Complexity (ECCC), 2007.

B. von Stengel.
Computing equilibria for two-person games.
Handbook of Game Theory, 3, 2002.