Algorithms for solving two-player normal form games
Recall: Nash equilibrium

- Let $A$ and $B$ be $|M| \times |N|$ matrices.
- Mixed strategies: Probability distributions over $M$ and $N$
- If player 1 plays $x$, and player 2 plays $y$, the payoffs are $x^T Ay$ and $x^T By$
- Given $y$, player 1’s best response maximizes $x^T Ay$
- Given $x$, player 2’s best response maximizes $x^T By$
- $(x,y)$ is a Nash equilibrium if $x$ and $y$ are best responses to each other
Finding Nash equilibria

- **Zero-sum games**
  - Solvable in poly-time using linear programming

- **General-sum games**
  - PPAD-complete
  - Several algorithms with exponential worst-case running time
    - Porter-Nudelman-Shoham [AAAI-04] = support enumeration
    - Sandholm-Gilpin-Conitzer [2005] - MIP Nash = mixed integer programming approach
Zero-sum games

- Among all best responses, there is always at least one pure strategy.

- Thus, player 1’s optimization problem is:

\[
\begin{align*}
\text{maximize} & \quad \min_{j \in N} \sum_{i \in M} a_{ij} x_i \\
\text{such that} & \quad \sum_{i \in M} x_i = 1 \\
& \quad x_i \geq 0 \text{ for all } i \in M
\end{align*}
\]

- This is equivalent to:

\[
\begin{align*}
\text{maximize} & \quad z \\
\text{such that} & \quad z - \sum_{i \in M} a_{ij} x_i \leq 0 \text{ for all } j \in N \\
& \quad \sum_{i \in M} x_i = 1 \\
& \quad x_i \geq 0 \text{ for all } i \in M
\end{align*}
\]

- By LP duality, player 2’s optimal strategy is given by the dual variables.
General-sum games: Lemke-Howson algorithm

- = pivoting algorithm similar to simplex algorithm
- We say each mixed strategy is “labeled” with the player’s unplayed pure strategies and the pure best responses of the other player
- A Nash equilibrium is a completely labeled pair (i.e., the union of their labels is the set of pure strategies)
Lemke-Howson Illustration

Example of label definitions

\[ A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix} \]
Lemke-Howson Illustration

Equilibrium 1

\[ A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix} \]
Lemke-Howson Illustration
Equilibrium 2

\[
A = \begin{bmatrix}
0 & 6 \\
2 & 5 \\
3 & 3 \\
\end{bmatrix}, \quad B = \begin{bmatrix}
1 & 0 \\
0 & 2 \\
4 & 3 \\
\end{bmatrix}
\]
Lemke-Howson Illustration
Equilibrium 3

\[ A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix} \]
Lemke-Howson Illustration

Run of the algorithm

\[ A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix} \]
Lemke-Howson Illustration

\[ A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix} \]
Lemke-Howson Illustration

\[
A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix}
\]
Lemke-Howson Illustration

\[ A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix} \]
Lemke-Howson Illustration

\[ A = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{bmatrix} \]
A subroutine that we’ll need when searching over supports

**Feasibility Program** (Checks whether there is a NE with given supports)

**Input:** $S = (S_1, \ldots, S_n)$, a support profile

**Output:** NE $p$, if there exists both a strategy profile $p = (p_1, \ldots, p_n)$ and a value profile $v = (v_1, \ldots, v_n)$ s.t.:

$$ \forall i \in N, a_i \in S_i : \sum_{a_{-i} \in S_{-i}} p(a_{-i})u_i(a_i, a_{-i}) = v_i $$

$$ \forall i \in N, a_i \notin S_i : \sum_{a_{-i} \in S_{-i}} p(a_{-i})u_i(a_i, a_{-i}) \leq v_i $$

$$ \forall i \in N : \sum_{a_i \in S_i} p_i(a_i) = 1 $$

$$ \forall i \in N, a_i \in S_i : p_i(a_i) \geq 0 $$

$$ \forall i \in N, a_i \notin S_i : p_i(a_i) = 0 $$

Solvable by LP
Features of PNS = support enumeration algorithm

- Separately instantiate supports
  - for each pair of supports, test whether there is a NE with those supports (using Feasibility Problem solved as an LP)
  - To save time, don’t run the Feasibility Problem on supports that include conditionally dominated actions
    - An $a_i$ is conditionally dominated, given $R_{-i} \subseteq A_{-i}$ if:

$$\exists a'_i \in A_i \ \forall a_{-i} \in R_{-i} : \ u_i(a_i, a_{-i}) < u_i(a'_i, a_{-i})$$

- Prefer balanced (= equal-sized for both players) supports
  - Motivated by a theorem: any nondegenerate game has a NE with balanced supports

- Prefer small supports
  - Motivated by existing theoretical results for particular distributions (e.g., [MB02])
Pseudocode of two-player PNS algorithm

for all support size profiles $x = (x_1, x_2)$, sorted in increasing order of, first, $|x_1 - x_2|$ and, second, $(x_1 + x_2)$ do
  for all $S_1 \subseteq A_1$ s.t. $|S_1| = x_1$ do
    $A'_2 \leftarrow \{ a_2 \in A_2 \text{ not cond. dominated, given } S_1 \}$
    if $\nexists a_1 \in S_1$ cond. dominated, given $A'_2$ then
      for all $S_2 \subseteq A'_2$ s.t. $|S_2| = x_2$ do
        if $\nexists a_1 \in S_1$ cond. dominated, given $S_2$ then
          if Feasibility Program 1 is satisfiable for $S = (S_1, S_2)$ then
            Return the found NE $p$
PNS: Experimental Setup

- Most previous empirical tests only on “random” games:
  - Each payoff drawn independently from uniform distribution
- GAMUT distributions [NWSL04]
  - Based on extensive literature search
  - Generates games from a wide variety of distributions
- Available at http://gamut.stanford.edu

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bertrand Oligopoly</td>
<td>2</td>
<td>Bidirectional LEG, Complete Graph</td>
</tr>
<tr>
<td>3</td>
<td>Bidirectional LEG, Random Graph</td>
<td>4</td>
<td>Bidirectional LEG, Star Graph</td>
</tr>
<tr>
<td>5</td>
<td>Covariance Game: $\rho = 0.9$</td>
<td>6</td>
<td>Covariance Game: $\rho = 0$</td>
</tr>
<tr>
<td>7</td>
<td>Covariance Game: Random $\rho^2 \in [-1/(N-1),1]$</td>
<td>8</td>
<td>Dispersion Game</td>
</tr>
<tr>
<td>9</td>
<td>Graphical Game, Random Graph</td>
<td>10</td>
<td>Graphical Game, Road Graph</td>
</tr>
<tr>
<td>11</td>
<td>Graphical Game, Star Graph</td>
<td>12</td>
<td>Location Game</td>
</tr>
<tr>
<td>13</td>
<td>Minimum Effort Game</td>
<td>14</td>
<td>Polymatrix Game, Random Graph</td>
</tr>
<tr>
<td>15</td>
<td>Polymatrix Game, Road Graph</td>
<td>16</td>
<td>Polymatrix Game, Small-World Graph</td>
</tr>
<tr>
<td>17</td>
<td>Random Game</td>
<td>18</td>
<td>Traveler’s Dilemma</td>
</tr>
<tr>
<td>19</td>
<td>Uniform LEG, Complete Graph</td>
<td>20</td>
<td>Uniform LEG, Random Graph</td>
</tr>
<tr>
<td>21</td>
<td>Uniform LEG, Star Graph</td>
<td>22</td>
<td>War Of Attrition</td>
</tr>
</tbody>
</table>
PNS: Experimental results on 2-player games

- Tested on 100 2-player, 300-action games for each of 22 distributions
- Capped all runs at 1800s
Mixed-Integer Programming
Methods for Finding Nash Equilibria

Tuomas Sandholm, Andrew Gilpin, Vincent Conitzer

[AAAI-05]
Motivation of MIP Nash

• Regret of pure strategy $s_i$ is difference in utility between playing optimally (given other player’s mixed strategy) and playing $s_i$.

• Observation: In any equilibrium, every pure strategy either is not played or has zero regret.

• Conversely, any strategy profile where every pure strategy is either not played or has zero regret is an equilibrium.
MIP Nash formulation

• For every pure strategy $s_i$:
  – There is a 0-1 variable $b_{s_i}$ such that
    • If $b_{s_i} = 1$, $s_i$ is played with 0 probability
    • If $b_{s_i} = 0$, $s_i$ is played with positive probability, but it must have 0 regret
  – There is a $[0,1]$ variable $p_{s_i}$ indicating the probability placed on $s_i$
  – There is a variable $u_{s_i}$ indicating the utility from playing $s_i$
  – There is a variable $r_{s_i}$ indicating the regret from playing $s_i$

• For each player $i$:
  – There is a variable $u_i$ indicating the utility player $i$ receives
  – There is a constant that captures the diff between her max and min utility:

$$U_i = \max_{s_i^h, s_i^l \in S_i, s_{1-i}^h, s_{1-i}^l \in S_{1-i}} u_i(s_i^h, s_{1-i}^h) - u_i(s_i^l, s_{1-i}^l)$$
MIP Nash formulation:
Only equilibria are feasible

\[
\begin{align*}
\text{find } p_{s_i}, u_i, u_{s_i}, r_{s_i}, b_{s_i} \text{ such that } & \\
(\forall i) \sum_{s_i \in S_i} p_{s_i} &= 1 \\
(\forall i)(\forall s_i \in S_i) \quad u_{s_i} &= \sum_{s_{1-i} \in S_{1-i}} p_{s_{1-i}} u_i(s_i, s_{1-i}) \\
(\forall i)(\forall s_i \in S_i) \quad u_i &\geq u_{s_i} \\
(\forall i)(\forall s_i \in S_i) \quad r_{s_i} &= u_i - u_{s_i} \\
(\forall i)(\forall s_i \in S_i) \quad p_{s_i} &\leq 1 - b_{s_i} \\
(\forall i)(\forall s_i \in S_i) \quad r_{s_i} &\leq U_i b_{s_i} \\
\text{domains: } p_{s_i} \geq 0, u_i \geq 0, u_{s_i} \geq 0, r_{s_i} \geq 0, b_{s_i} \in \{0, 1\}. 
\end{align*}
\]
MIP Nash formulation:
Only equilibria are feasible

- Has the advantage of being able to specify objective function
  - Can be used to find optimal equilibria (for any linear objective)
MIP Nash formulation

- Other three formulations explicitly make use of regret minimization:
  Formulation 2. Penalize regret on strategies that are played with positive probability
  Formulation 3. Penalize probability placed on strategies with positive regret
  Formulation 4. Penalize either the regret of, or the probability placed on, a strategy
MIP Nash: Comparing formulations

These results are from a newer, extended version of the paper.

<table>
<thead>
<tr>
<th></th>
<th>Form. 1</th>
<th>Form. 2</th>
<th>Form. 3</th>
<th>Form. 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>BertrandOligopoly</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>BidirectionalLEG.CG</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>BidirectionalLEG.RG</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>BidirectionalLEG.SG</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CovariantGame.Pos</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CovariantGame_Rand</td>
<td>864</td>
<td>864</td>
<td>844.56</td>
<td>864</td>
</tr>
<tr>
<td>CovariantGame_Zero</td>
<td>300.97</td>
<td>432</td>
<td>172.01</td>
<td>432</td>
</tr>
<tr>
<td>DispersionGame</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>GraphicalGame_RG</td>
<td>289.41</td>
<td>576</td>
<td>270.89</td>
<td>576</td>
</tr>
<tr>
<td>GraphicalGame_Road</td>
<td>440.34</td>
<td>648</td>
<td>293.69</td>
<td>648</td>
</tr>
<tr>
<td>GraphicalGame_SG</td>
<td>428.82</td>
<td>720</td>
<td>216.46</td>
<td>720</td>
</tr>
<tr>
<td>GraphicalGame_SW</td>
<td>433.31</td>
<td>864</td>
<td>279.02</td>
<td>864</td>
</tr>
<tr>
<td>LocationGame</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MinimumEffortGame</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PolymatrixGame.CG</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
</tr>
<tr>
<td>PolymatrixGame_RG</td>
<td>36.04</td>
<td>72</td>
<td>32.84</td>
<td>72</td>
</tr>
<tr>
<td>PolymatrixGame_Road</td>
<td>237.21</td>
<td>360</td>
<td>217.93</td>
<td>360</td>
</tr>
<tr>
<td>PolymatrixGame_SW</td>
<td>95</td>
<td>216</td>
<td>101.53</td>
<td>216</td>
</tr>
<tr>
<td>RandomGame</td>
<td>515.27</td>
<td>1008</td>
<td>411.3</td>
<td>1008</td>
</tr>
<tr>
<td>TravelersDilemma</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>UniformLEG.CG</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>UniformLEG.RG</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>UniformLEG.SG</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>WarOfAttrition</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>OVERALL:</strong></td>
<td><strong>3712.37</strong></td>
<td><strong>5832</strong></td>
<td><strong>2912.23</strong></td>
<td><strong>5832</strong></td>
</tr>
</tbody>
</table>

Table 1: Average time (in seconds) to find an equilibrium using the different MIP formulations, in 150 x 150 games from the GAMUT distributions (25 instances of each). If an instance reached the 1800 second limit, that time was counted toward the average. Zero entries indicate that a pure strategy equilibrium existed in each of the 25 instances.
Games with medium-sized supports

• Since PNS performs support enumeration, it should perform poorly on games with medium-sized support

• There is a family of games such that there is a single equilibrium, and the support size is about half
  – And, none of the strategies are dominated (no cascades either)
MIP Nash: Computing optimal equilibria

- MIP Nash is best at finding *optimal* equilibria
- Lemke-Howson and PNS are good at finding sample equilibria
  - M-Enum is an algorithm similar to Lemke-Howson for enumerating all equilibria
- M-Enum and PNS can be modified to find optimal equilibria by finding all equilibria, and choosing the best one
  - In addition to taking exponential time, there may be exponentially many equilibria

<table>
<thead>
<tr>
<th>actions</th>
<th>M-Enum</th>
<th>PNS</th>
<th>MIP Nash</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2.21 (0%)</td>
<td>26.45 (3.7%)</td>
<td>0.001 (0%)</td>
</tr>
<tr>
<td>25</td>
<td>429.14 (66.7%)</td>
<td>600 (100%)</td>
<td>6.97 (0%)</td>
</tr>
<tr>
<td>50</td>
<td>425.07 (66.7%)</td>
<td>600 (100%)</td>
<td>27.2 (2.1%)</td>
</tr>
</tbody>
</table>

Table 3: Average time (in seconds), over all GAMUT distributions (6 instances of each), for finding a welfare-maximizing equilibrium. The percentage of timeouts (limit here was 600s) is in parentheses.
Algorithms for solving other types of games
Structured games

• Graphical games
  – Payoff to $i$ only depends on a subset of the other agents
  – Poly-time algorithm for undirected trees (Kearns, Littman, Singh 2001)
  – Graphs (Ortiz & Kearns 2003)
  – Directed graphs (Vickery & Koller 2002)

• Action-graph games (Bhat & Leyton-Brown 2004)
  – Each agent’s action set is a subset of the vertices of a graph
  – Payoff to $i$ only depends on number of agents who take neighboring actions
Games with more than two players

• For finding a Nash equilibrium
  – Problem is no longer a linear complementarity problem
    • So Lemke-Howson does not apply
  – Simplicial subdivision
    • Path-following method derived from Scarf’s algorithm
    • Exponential in worst-case
  – Govindan-Wilson
    • Continuation-based method
    • Can take advantage of structure in games
  – Non globally convergent methods (*i.e.* incomplete)
    • Non-linear complementarity problem
    • Minimizing a function
    • Slow in practice

• What about strong Nash equilibrium or coalition-proof Nash equilibrium?