Computing Solution Concepts – Intro

Solution Concepts

- In single agent settings, there is the notion of *optimal strategy*
- In multiagent setting, situation is more complex. Best strategy depends on the strategies of other agents
- *Solution concepts* – certain subsets of outcomes that are interesting
- Pareto optimality, Nash equilibrium

Computational Concerns

- How to compute a Nash equilibrium?
- Examples we've seen had 2 players, each with 2 actions
- Complexity depends on class of games considered
- 2 player zero-sum games
- 2 player general-sum games
- n players, n > 2
- Other solution concepts

Computing Nash Equilibrium in 2-player, zero-sum games

Setup

- Consider 2-player, zero-sum game:

$$G = (\{1, 2\}, A_1 \times A_2, (u_1, u_2))$$

- Let U^*_i be the equilibrium value of player I
- Recall in a N.E., player 1's value is equal to his maxmin value

Definition 3.4.1 (Maxmin) *The* maxmin strategy *for player i is* $\arg\max_{s_i} \min_{s_{-i}} u_i(s_i, s_{-i})$, *and the* maxmin value *for player i is* $\max_{s_i} \min_{s_{-i}} u_i(s_i, s_{-i})$.

- Use this fact to write an LP

Linear Program

$$
\begin{aligned}
&\text{minimize} && U_1^* && (4.1)\\
&\text{subject to} && \sum_{k \in A_2} u_1(a_1^j, a_2^k) \cdot s_2^k \leq U_1^* \qquad \forall j \in A_1 && (4.2)\\
& && \sum_{k \in A_2} s_2^k = 1 && (4.3)\\
& && s_2^k \geq 0 \qquad \forall k \in A_2 && (4.4)
\end{aligned}
$$

- Variables: U^*_i, $s_2{}^k$

Dual Program

$$\text{maximize} \quad U_1^* \tag{4.5}$$

$$\text{subject to} \quad \sum_{j \in A_1} u_1(a_1^j, a_2^k) \cdot s_1^j \geq U_1^* \qquad \forall k \in A_2 \tag{4.6}$$

$$\sum_{j \in A_1} s_1^j = 1 \tag{4.7}$$

$$s_1^j \geq 0 \qquad \forall j \in A_1 \tag{4.8}$$

Reformulation with Slack Variables

$$\text{minimize} \quad U_1^* \tag{4.9}$$

$$\text{subject to} \quad \sum_{k \in A_2} u_1(a_1^j, a_2^k) \cdot s_2^k + r_1^j = U_1^* \qquad \forall j \in A_1 \tag{4.10}$$

$$\sum_{k \in A_2} s_2^k = 1 \tag{4.11}$$

$$s_2^k \geq 0 \qquad \forall k \in A_2 \tag{4.12}$$

$$r_1^j \geq 0 \qquad \forall j \in A_1 \tag{4.13}$$

Example: Computing Nash Equilibrium in Mixed Pennies

Game Setup: Matching Pennies

Payoffs to P1:

$$U1=\begin{bmatrix} 1 & -1, \\ -1 & 1 \end{bmatrix}, \quad U2=-U1$$

- If coins match → P1 wins (+1), P2 loses (−1)
- If coins differ → P2 wins (+1), P1 loses (−1)

Mixed Strategies

Let:

- P1 plays H with probability p, T with probability 1–p
- P2 plays H with probability q, T with probability 1–q
- We'll solve for (p,q) that form a Nash equilibrium.

Indifference Condition, P1

Expected payoff to P1 if they play H:

- $u_1(H, q) = 1 \cdot q + (-1) \cdot (1-q) = 2q - 1$

  Expected payoff if they play T:

- $u_1(T, q) = (-1) \cdot q + 1 \cdot (1-q) = 1 - 2q$

  Indifference requirement:

- $u_1(H, q) = u_1(T, q)$
- $\Rightarrow 2q - 1 = 1 - 2q \Rightarrow q = 1/2$

Indifference Condition, P2

- By symmetry, if P1 plays H with probability p:

- $u_2(H, p) = 1 - 2p$
- $u_2(T, p) = 2p - 1$

- Indifference:
- $1 - 2p = 2p - 1 \Rightarrow p = 1/2$

  So P1 must also mix equally.

Equilibrium Summary

- Nash Equilibrium: $p = 1/2$, $q = 1/2$

- Each player randomizes equally between H and T.
- Expected payoff = 0 to both players.

Interpretation:

• Matching Pennies illustrates the need for mixed strategies.

• Randomization prevents exploitation.

Maxmin LP for Matching Pennies

- We can solve for Player 1's equilibrium strategy via a linear program.
- Let variables: $p_H, p_T \geq 0$ (probabilities for Heads, Tails). v = guaranteed payoff (value of the game)
- Constraints (for each column j of U_1):
- $\sum_i p_i \cdot U_1[i,j] \geq v$

  (ensures expected payoff ≥ v against each pure strategy of P2)

Maxmin LP for Matching Pennies

- Constraints (for each column j of U_1):
- $\sum_i p_i \cdot U_1[i,j] \geq v$
- Also: $p_H + p_T = 1$
- Objective: maximize v
- For Matching Pennies, this yields: $p_H = 1/2$, $p_T = 1/2$, $v = 0$

Complexity of Computing a Nash Equilibrium

Computing Nash Equilibria

- 2-player, zero-sum in poly
- 2-player, general sum?
- Cannot be formulated as LP, players not diametrically opposed
- No known reduction from NP-complete problem
- Stumbling block with NP: decision problems. But we always know a NE exists

The PPAD class

- So current knowledge about NE computation is in relation to PPAD class
- PPAD – "Polynomial Parity Argument, Directed Version"
- Family of directed graphs *G(n)*
- Computational task is finding a source or sink node

The family *G(n)*

- Defined on set N of 2^n nodes, but described in polynomial space
- Just encode set of edges
- *Parent, Child* functions from N to N: encoded as arithmetic circuits with sizes poly. in n
- An edge exists from node j to k iff. $Parent(k) = j$ and $Child(j) = k$
- There must exist one distinguished node 0 with exactly zero parents
- Find sink or source other than 0 in a given graph

Complexity

Theorem 4.2.1 *The problem of finding a sample Nash equilibrium of a general-sum finite game with two or more players is PPAD-complete.*

- CNE is in PPAD and any other problem in PPAD can be reduced to it
- CNE is in PPAD reduction proceeds quite directly from the proof in the textbook that every game has a NE that uses Sperner's lemma
- Harder part is showing CNE is PPAD-hard. Result was proven in 2005, a culmination of intermediate results achieved over a decade

Complexity

Theorem 4.2.1 *The problem of finding a sample Nash equilibrium of a general-sum finite game with two or more players is PPAD-complete.*

- Not known if P=PPAD. Generally believed not
- It is known that finding an NE in 2 player games is no easier than finding an NE in n player games
- Finding a NE is no easier than finding an arbitrary Brouwer fixed point

LCP Formulation of 2-player NE

Computing Nash Equilibria

- 2-player, zero-sum in poly
- 2-player, general sum?
- Cannot be formulated as LP, players not diametrically opposed
- The LCP formulation

The LCP Formulation

$$\sum_{k\in A_2} u_1(a_1^j, a_2^k)\cdot s_2^k + r_1^j = U_1^* \qquad \forall j \in A_1 \tag{4.14}$$

$$\sum_{j\in A_1} u_2(a_1^j, a_2^k)\cdot s_1^j + r_2^k = U_2^* \qquad \forall k \in A_2 \tag{4.15}$$

$$\sum_{j\in A_1} s_1^j = 1, \quad \sum_{k\in A_2} s_2^k = 1 \tag{4.16}$$

$$s_1^j \geq 0, \quad s_2^k \geq 0 \qquad \forall j \in A_1, \forall k \in A_2 \tag{4.17}$$

$$r_1^j \geq 0, \quad r_2^k \geq 0 \qquad \forall j \in A_1, \forall k \in A_2 \tag{4.18}$$

$$r_1^j \cdot s_1^j = 0, \quad r_2^k \cdot s_2^k = 0 \qquad \forall j \in A_1, \forall k \in A_2 \tag{4.19}$$

The Lemke-Howson Algorithm

Lemke-Howson Algorithm

- 2-player, general sum games
- Algorithm is for solving linear complementarity programs
- Searches vertices of strategy simplices (like the simplex algorithm for solving LPs)

Lemke-Howson – a graphical exposition

0, 1	6, 0
2, 0	5, 2
3, 4	3, 3

Figure 4.1: A game for the exposition of the Lemke–Howson algorithm.

Lemke-Howson – a graphical exposition

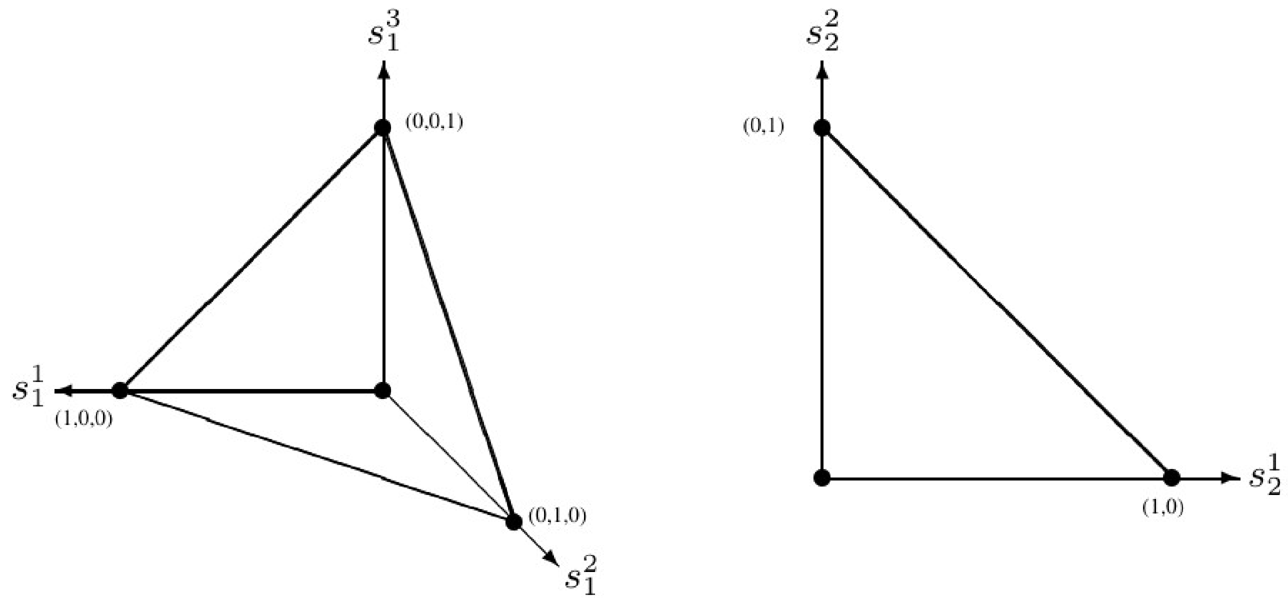


Figure 4.2: Strategy spaces for player 1 (left) and player 2 (right) in the game from Figure 4.1.

Lemke-Howson – a graphical exposition

Our next step in defining the Lemke–Howson algorithm is to define a labeling on the strategies. Every possible mixed strategy s_i is given a set of labels $L(s_i^j) \subseteq A_1 \cup A_2$ drawn from the set of available actions for both players. Denoting a given player as i and the other player as $-i$, mixed strategy s_i for player i is labeled as follows:

- with each of player i's actions a_i^j that is *not* in the support of s_i; and
- with each of player $-i$'s actions a_{-i}^j that *is* a best response by player $-i$ to s_i.

Lemke-Howson – a graphical exposition

Our next step in defining the Lemke–Howson algorithm is to define a labeling on the strategies. Every possible mixed strategy s_i is given a set of labels $L(s_i^j) \subseteq A_1 \cup A_2$ drawn from the set of available actions for both players. Denoting a given player as i and the other player as $-i$, mixed strategy s_i for player i is labeled as follows:

- with each of player i's actions a_i^j that is *not* in the support of s_i; and
- with each of player $-i$'s actions a_{-i}^j that *is* a best response by player $-i$ to s_i.

- A strategy profile is a Nash equilibrium iff. it is completely labeled

Lemke-Howson – a graphical exposition

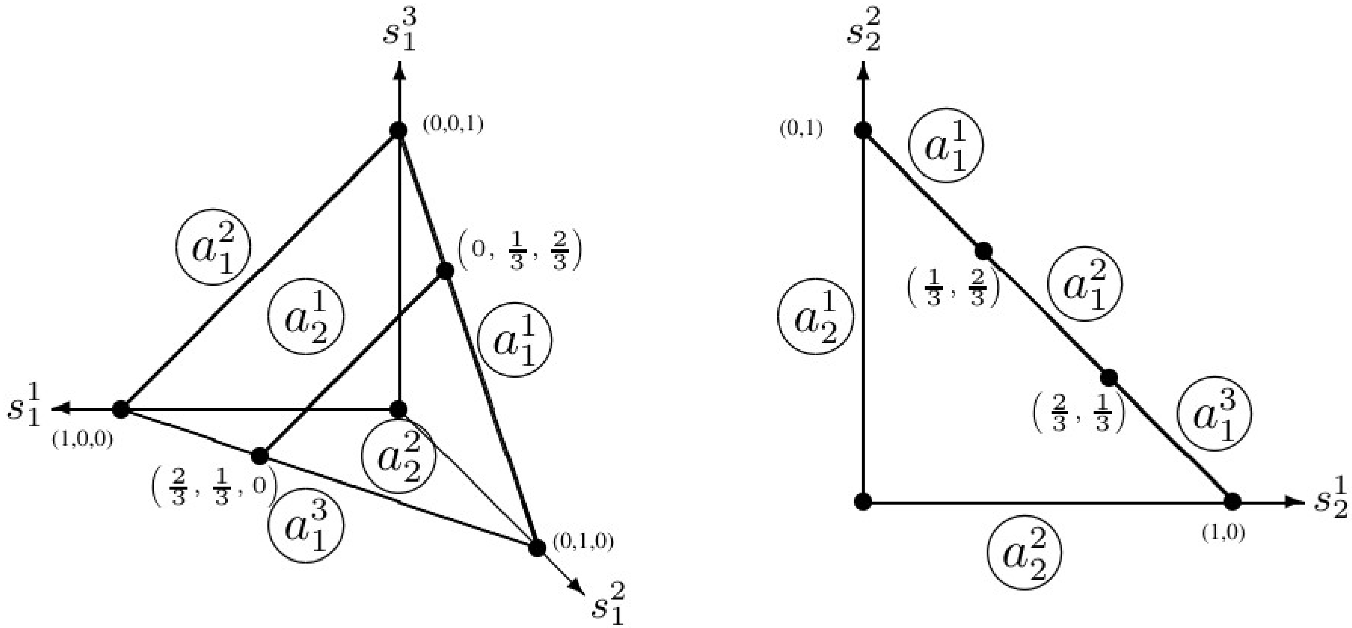
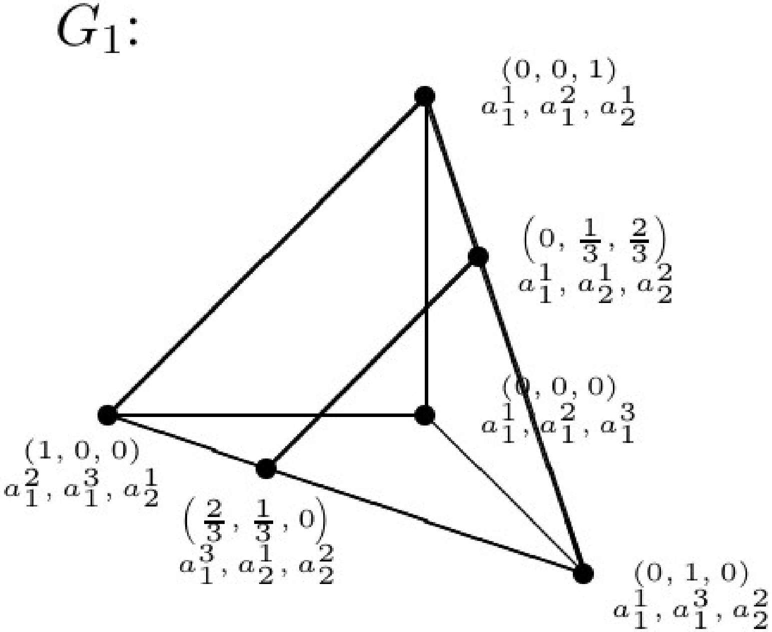
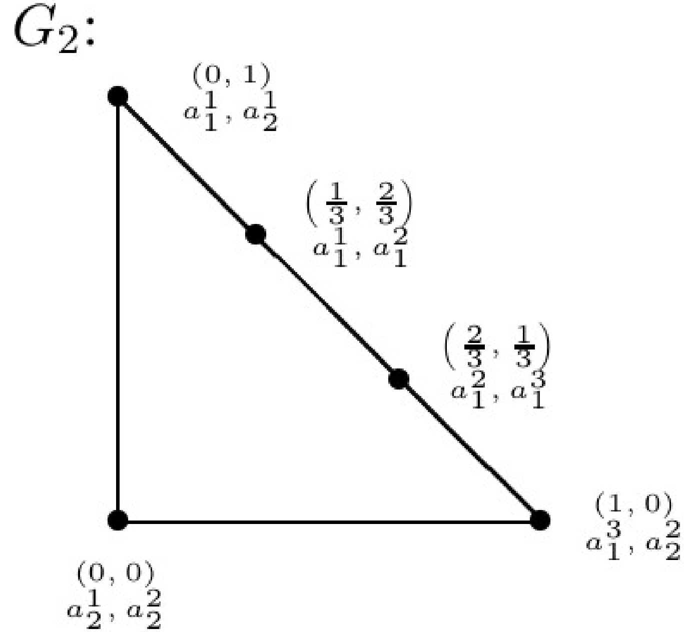


Figure 4.3: Labeled strategy spaces for player 1 (left) and player 2 (right) in the game from Figure 4.1.

Lemke-Howson – a graphical exposition





Lemke-Howson – Properties

- Guaranteed to find a NE
- Alternative proof of the existence of NE
- Path after initial move is unique. Only nondeterminism is in first move
- All paths from the starting point to a NE can be exponential (algorithm is provably exponential)
- No way to assess how close we are to a NE