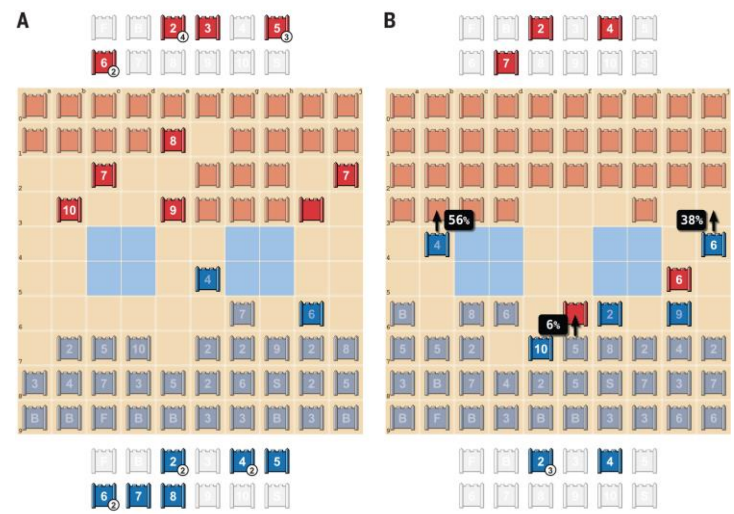
Imperfect-Information Games 2

Stephen McAleer

Sequential Decision-Making

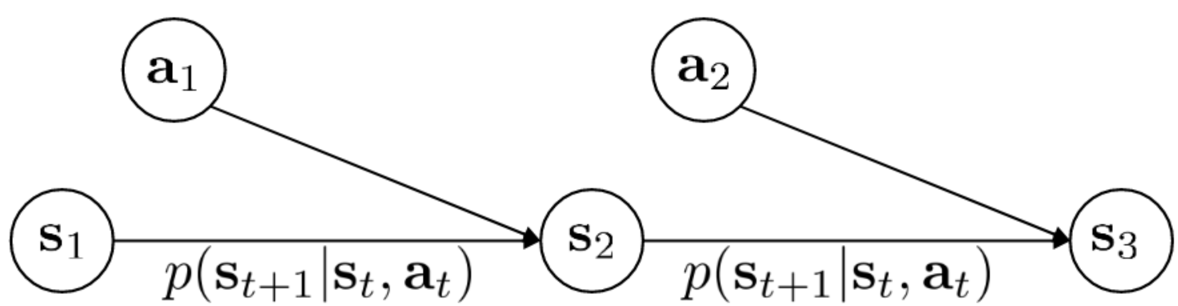
- Most real-world games involve sequential decision making
- Actions affect the future
- Low-level control can become a significant hurdle
  - E.g. Robotics
- Emergence of bluffing, deception



Single-Agent Setting: Markov Decision Process (MDP)

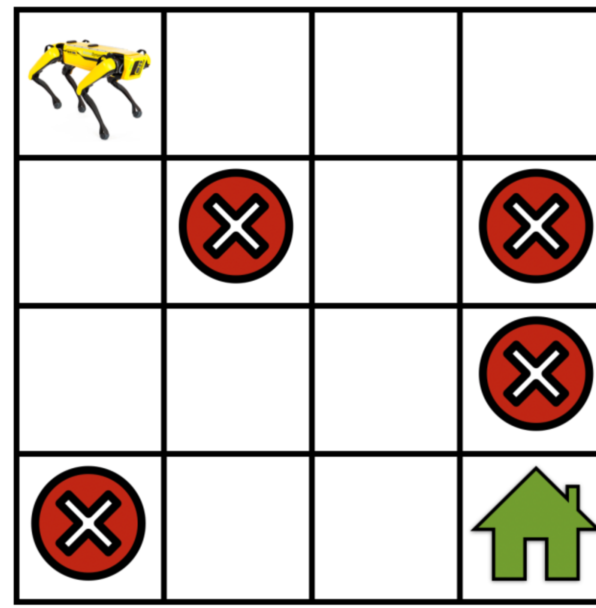
A Markov Decision Process is formally defined as a tuple $(S, A, P_{sa}, \gamma, R)$ where:

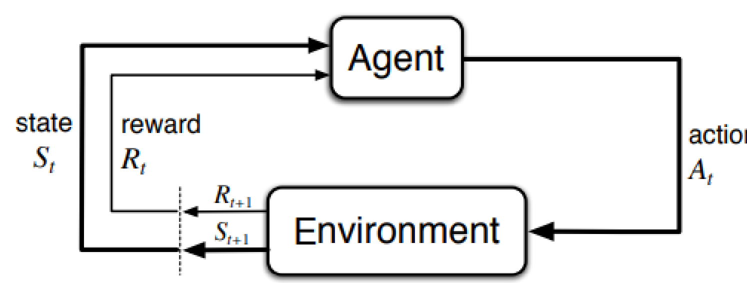
- S is a finite set of states.
- A is a finite set of actions.
- $P_{sa}(s') = \mathbb{P}(S_{t+1} = s' | S_t = s, A_t = a)$ is the transition probability that action a in state s at time t will lead to state s' at time $t+1$.
- γ is the discount factor, $0 \leq \gamma \leq 1$.
- R is a reward function, which can be state-dependent $R(s)$ or state-action dependent $R(s, a)$.



MDP Example

- States
  - Each cell
- Action Space
  - Up, down, left, right
- Transition Probabilities
  - Go in action direction with prob 1-ε
  - Go in random direction with prob ε
- Discount Factor
  - 0.99
- Reward Function
  - 0 on normal cells
  - 1 on goal cell
  - -1 on red cells





Goal of Reinforcement Learning

- A policy provides a mapping from states to actions
  - Could be distribution over actions

$$\pi : S \to A$$

- The value function returns the expected value of a policy at a particular state

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid s_0 = s, \pi\right]$$

Goal of Reinforcement Learning

- A policy provides a mapping from states to actions
  - Could be distribution over actions

$$\pi : S \to A$$

- The value function returns the expected value of a policy at a particular state

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid s_0 = s, \pi\right]$$

- Objective is to find the optimal policy

$$\pi^* = \arg\max_\pi \mathbb{E}_{s_0 \sim p(s_0)}[V^\pi(s_0)]$$

Bellman Equations

- Value function can be recursively defined

$$V^{\pi}(s) = R(s) + \gamma \sum_{s' \in S} P_{sa}(s') V^{\pi}(s')$$

Bellman Equations

- Value function can be recursively defined

$$V^{\pi}(s) = R(s) + \gamma \sum_{s' \in S} P_{sa}(s') V^{\pi}(s')$$

- Q values return expected value for state and action

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^{t} R(s_t, a_t) \mid s_0 = s, a_0 = a, \pi\right]$$

Bellman Equations

- Value function can be recursively defined

$$V^{\pi}(s) = R(s) + \gamma \sum_{s' \in S} P_{sa}(s') V^{\pi}(s')$$

- Q values return expected value for state and action

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 = s, a_0 = a, \pi\right]$$

- and also can be defined recursively

$$Q^{\pi}(s, a) = R(s, a) + \gamma \sum_{s' \in S} P_{sa}(s') \sum_{a' \in A} \pi(a'|s') Q^{\pi}(s', a')$$

Q-Learning

- Maintain a table of Q-values for each state-action pair
- Iteratively update this table via bootstrapped target until convergence
- Improvement comes from the max operator

$$\underbrace{Q(S_t, A_t)}_{\text{New Q-value estimation}} \leftarrow \underbrace{Q(S_t, A_t)}_{\text{Former Q-value estimation}} + \underbrace{\alpha}_{\text{Learning Rate}}[\underbrace{\underbrace{\underbrace{R_{t+1}}_{\text{Immediate Reward}} + \underbrace{\gamma max_a Q(S_{t+1}, a)}_{\text{Discounted Estimate optimal Q-value of next state}}}_{\text{TD Target}} - \underbrace{Q(S_t, A_t)}_{\text{Former Q-value estimation}}}_{\text{TD Error}}]$$

Markov Games

- Like MDP, but multiple agents give actions

$$(S, \{A_i\}, \{R_i\}, P, \gamma)$$

- When only one state, same as normal-form game
- Essentially each timestep you play a normal-form game then everyone transitions to the next state

Markov Games

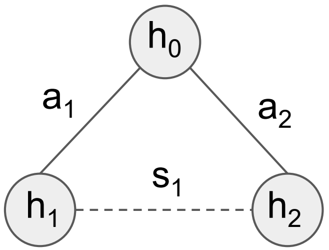
- Like MDP, but multiple agents give simultaneous actions

$$(S, \{A_i\}, \{R_i\}, P, \gamma)$$

- When only one state, same as normal-form game
- Essentially each timestep you play a normal-form game then everyone transitions to the next state
- Due to Markov assumption, doesn't model many (most?) real-world or game settings such as poker or Stratego
  - Only hidden info is due to synchronous moves
- We want to generally model **imperfect information**

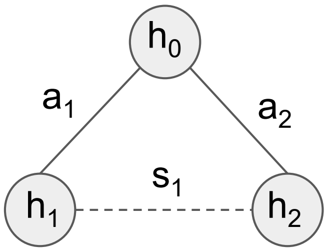
Extensive-Form Games

- ***History h*** is ground truth state of the game
  - All cards for all players
- ***Information set s*** is observation for one player
  - Set of histories consistent with observation
  - The hand for one player



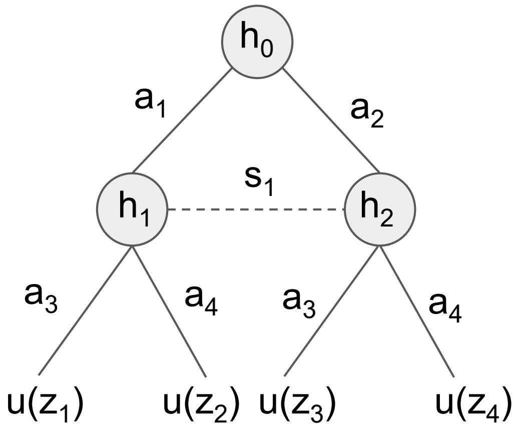
Extensive-Form Games

- ***History h*** is ground truth state of the game
    - All cards for all players
- ***Information set s*** is observation for one player
    - Set of histories consistent with observation
    - The hand for one player
- ***Strategy (policy) $\pi_i(a|s)$*** gives distribution over actions at information set s



Extensive-Form Games

- ***History h*** is ground truth state of the game
    - All cards for all players
- ***Information set s*** is observation for one player
    - Set of histories consistent with observation
    - The hand for one player
- ***Strategy (policy)*** $\boldsymbol{\pi_i(a|s)}$ gives distribution over actions at information set s
- ***Terminal history z*** is history at end of game
- ***Utility*** $\boldsymbol{u_i(z)}$ is utility for player i



Extensive-Form Games

- General way to model sequential games such as poker and Stratego
- Kind of like Partially-Observable Markov Games
  - Tree form (no looping)
  - Finite horizon
- Connection to reinforcement learning
  - Strategy = policy
  - Information set = set of all past observations
  - Utility = reward

Connection to Normal Form

- Normal -> extensive form
  - Player one acts first
  - Player two's information set includes all possible P1 actions

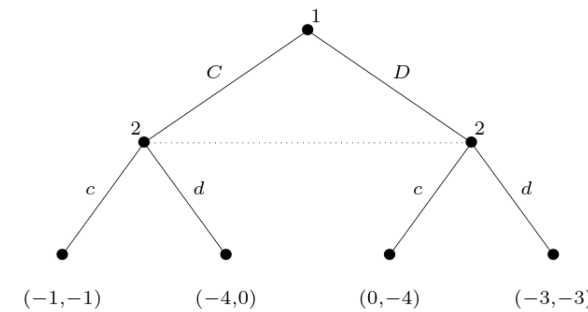


Figure 5.11: The Prisoner's Dilemma game in extensive form.

Connection to Normal Form

- Normal -> extensive form
  - Player one acts first
  - Player two’s information set includes all possible P1 actions

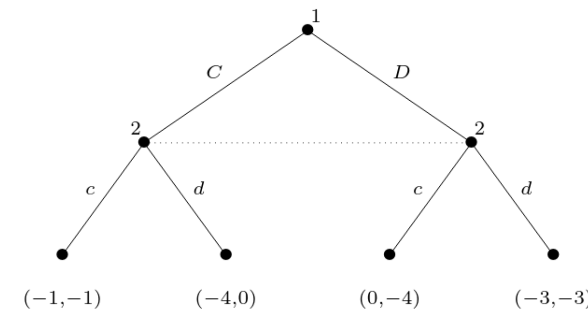
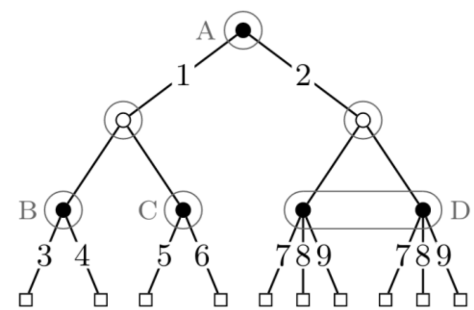


Figure 5.11: The Prisoner’s Dilemma game in extensive form.

- Extensive -> normal form
  - NF pure strategy picks one action for each information set
  - **Combinatorial blow-up**



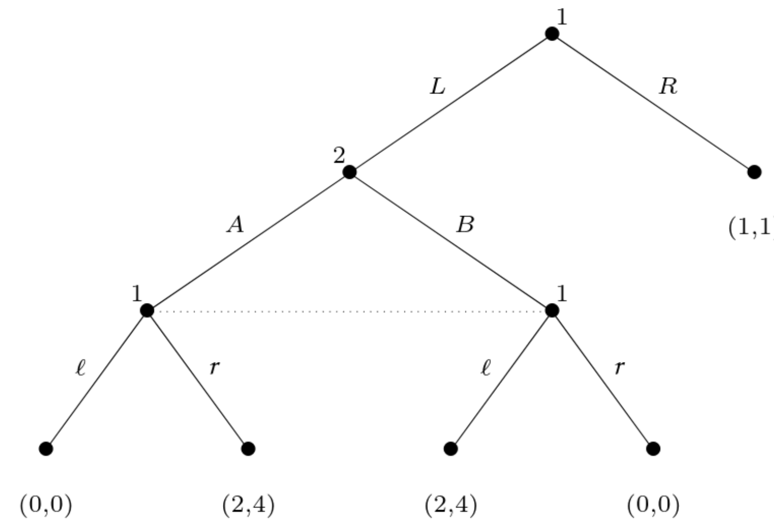
1357, 1358, 1359, 1367, 1368, 1369,
1457, 1458, 1459, 1467, 1468, 1469,
2357, 2358, 2359, 2367, 2368, 2369,
2457, 2458, 2459, 2467, 2468, 2469

Pure NF strategies for player 1

Kuhn's Theorem [1953] shows that mixed strategies and behavioral strategies are equivalent

Sequence Form

- Sequence σ: series of actions taken by a player to reach a history h
  - h may or may not be terminal
  - Doesn't include opponent actions



	$\emptyset$	A	B
$\emptyset$	0, 0	0, 0	0, 0
L	0, 0	0, 0	0, 0
R	1, 1	0, 0	0, 0
$L\ell$	0, 0	0, 0	2, 4
Lr	0, 0	2, 4	0, 0

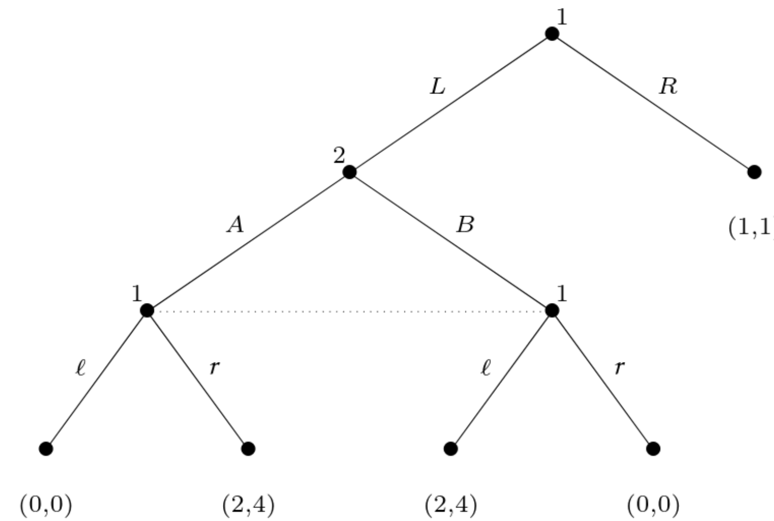
Figure 5.13: The sequence form of the game from Figure 5.10.

	A	B
$L\ell$	0, 0	2, 4
Lr	2, 4	0, 0
$R\ell$	1, 1	1, 1
Rr	1, 1	1, 1

Figure 5.14: The induced normal form of the game from Figure 5.10.

Sequence Form

- Sequence σ: series of actions taken by a player to reach a history h
  - h may or may not be terminal
  - Doesn't include opponent actions
- Payoff of two sequences is u(z) if the sequences lead to z, otherwise 0
- **Number of sequences linear in game tree size**



	$\emptyset$	A	B
$\emptyset$	0, 0	0, 0	0, 0
L	0, 0	0, 0	0, 0
R	1, 1	0, 0	0, 0
$L\ell$	0, 0	0, 0	2, 4
Lr	0, 0	2, 4	0, 0

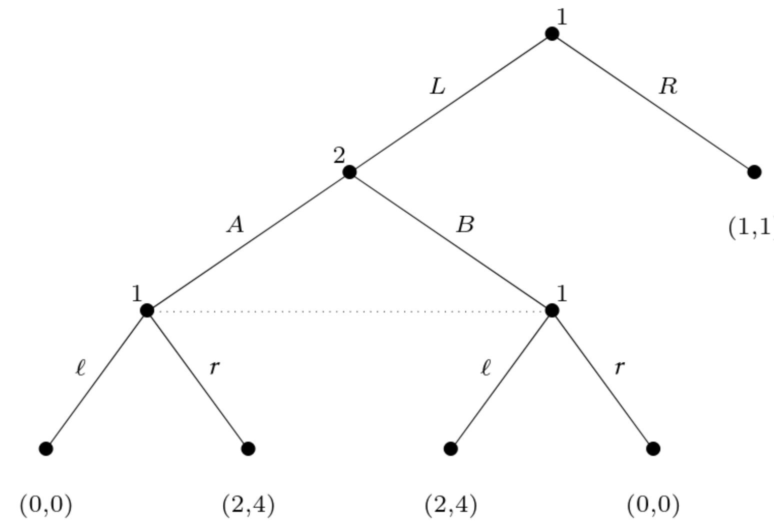
Figure 5.13: The sequence form of the game from Figure 5.10.

	A	B
$L\ell$	0, 0	2, 4
Lr	2, 4	0, 0
$R\ell$	1, 1	1, 1
Rr	1, 1	1, 1

Figure 5.14: The induced normal form of the game from Figure 5.10.

Sequence Form

- Sequence σ: series of actions taken by a player to reach a history h
  - h may or may not be terminal
  - Doesn't include opponent actions
- Payoff of two sequences is u(z) if the sequences lead to z, otherwise 0
- **Number of sequences linear in game tree size**
- What is a strategy (policy)?



	$\emptyset$	A	B
$\emptyset$	0, 0	0, 0	0, 0
L	0, 0	0, 0	0, 0
R	1, 1	0, 0	0, 0
$L\ell$	0, 0	0, 0	2, 4
Lr	0, 0	2, 4	0, 0

Figure 5.13: The sequence form of the game from Figure 5.10.

	A	B
$L\ell$	0, 0	2, 4
Lr	2, 4	0, 0
$R\ell$	1, 1	1, 1
Rr	1, 1	1, 1

Figure 5.14: The induced normal form of the game from Figure 5.10.

From Behavior Strategy to Sequences

- *Realization probability* of a sequence σ of player i under strategy π_i
  - Product of action probabilities needed to play sequence for player i

$$\pi_i[\sigma] = \prod_{a \in \sigma} \pi_i(a)$$

From Behavior Policy to Sequences

- *Realization probability* of a sequence σ of player i under strategy π_i
  - Product of action probabilities needed to play sequence for player i

$$\pi_i[\sigma] = \prod_{a \in \sigma} \pi_i(a)$$

- *Realization plan* provides probabilities for each sequence under strategy π

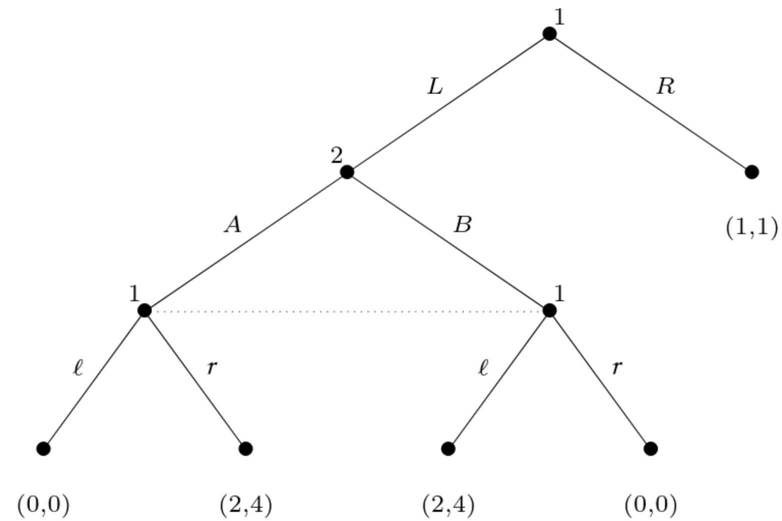
$$x(\sigma) = \pi_1[\sigma]$$
$$y(\sigma) = \pi_2[\sigma]$$

Realization Plan Example

- $\pi_1(L) = \frac{1}{2}$, $\pi_1(R) = \frac{1}{2}$,

  $\pi_1(l) = \frac{1}{3}$, $\pi_1(r) = \frac{2}{3}$
- $x(L) = \frac{1}{2}$, $x(R) = \frac{1}{2}$,

  $x(Ll) = \frac{1}{6}$, $x(Lr) = \frac{1}{3}$



	$\emptyset$	A	B
$\emptyset$	0, 0	0, 0	0, 0
L	0, 0	0, 0	0, 0
R	1, 1	0, 0	0, 0
$L\ell$	0, 0	0, 0	2, 4
Lr	0, 0	2, 4	0, 0

Figure 5.13: The sequence form of the game from Figure 5.10.

	A	B
$L\ell$	0, 0	2, 4
Lr	2, 4	0, 0
$R\ell$	1, 1	1, 1
Rr	1, 1	1, 1

Figure 5.14: The induced normal form of the game from Figure 5.10.

From Realization Plans to Policies

A realization plan x of a behavior strategy of player 1 fulfills

$$
\begin{aligned}
x(\sigma) &\geq 0 && \text{for all } \sigma \in S_1, \\
x(\emptyset) &= 1, \\
\sum_{a \in A_s} x(\sigma_s a) &= x(\sigma_s) && \text{for every information set } s \in S.
\end{aligned}
$$

Conversely, if a realization plan x fulfills these properties, it corresponds to a behavior strategy. The strategy prob for action a is just $x(\sigma a)/x(\sigma)$

From Realization Plans to Policies

A realization plan x of a behavior strategy of player 1 fulfills

$$
\begin{aligned}
x(\sigma) &\geq 0 && \text{for all } \sigma \in S_1, \\
x(\emptyset) &= 1, \\
\sum_{a \in A_s} x(\sigma_s a) &= x(\sigma_s) && \text{for every information set } s \in S.
\end{aligned}
$$

Conversely, if a realization plan x fulfills these properties, it corresponds to a behavior strategy. The strategy prob for action a is just $x(\sigma a)/x(\sigma)$

If x satisfies these constraints, it is in the *sequence-form polytope X*

Characterizing the Sequence-Form Polytope

- Can represent the previous constraints compactly

$$\mathcal{X} = \{\boldsymbol{x} : \mathbf{F}_1 \boldsymbol{x} = \boldsymbol{f}_1, \boldsymbol{x} \geq \mathbf{0}\}$$

Characterizing the Sequence-Form Polytope

- Can represent the previous constraints compactly

$$\mathcal{X} = \{\boldsymbol{x} : \mathbf{F}_1 \boldsymbol{x} = \boldsymbol{f}_1, \boldsymbol{x} \geq \mathbf{0}\}$$

- In our example, 5 sequences (Ø, L, R, Ll, Lr), so x is 5-dim vector

$$F_1 = \begin{bmatrix} 1 & & & & \\ -1 & 1 & 1 & & \\ & -1 & & 1 & 1 \end{bmatrix}$$

$$f_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

	$\emptyset$	A	B
$\emptyset$	$0, 0$	$0, 0$	$0, 0$
L	$0, 0$	$0, 0$	$0, 0$
R	$1, 1$	$0, 0$	$0, 0$
$L\ell$	$0, 0$	$0, 0$	$2, 4$
Lr	$0, 0$	$2, 4$	$0, 0$

Figure 5.13: The sequence form of the game from Figure 5.10.

Sequence-Form LP

- Goal is to solve this bilinear saddle point problem

$$\max_{\boldsymbol{x} \in \mathcal{X}} \min_{\boldsymbol{y} \in \mathcal{Y}} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y}$$

where *X* and *Y* are the sequence-form polytopes of the players

Sequence-Form LP

- Goal is to solve this bilinear saddle point problem

$$\max_{\boldsymbol{x}\in\mathcal{X}} \min_{\boldsymbol{y}\in\mathcal{Y}} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y}$$

where *X* and *Y* are the sequence-form polytopes of the players

- Let’s define the inner optimization objective

$$g(\boldsymbol{x}) := \min_{\boldsymbol{y}\in\mathcal{Y}} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y}$$

Sequence-Form LP

- Goal is to solve this bilinear saddle point problem

$$\max_{\boldsymbol{x}\in\mathcal{X}} \min_{\boldsymbol{y}\in\mathcal{Y}} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y}$$

where *X* and *Y* are the sequence-form polytopes of the players

- Let’s define the inner optimization objective

$$g(\boldsymbol{x}) := \min_{\boldsymbol{y}\in\mathcal{Y}} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y}$$

So that

$$\max_{\boldsymbol{x}\in\mathcal{X}} \min_{\boldsymbol{y}\in\mathcal{Y}} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y} = \max_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x})$$

Sequence-Form LP

- Can rewrite g(x) as a linear program

$$g(\boldsymbol{x}) = \begin{cases} \min & (\mathbf{A}^\top \boldsymbol{x})^\top \boldsymbol{y} \\ \text{s.t.} & ① \; \mathbf{F}_2 \boldsymbol{y} = \boldsymbol{f}_2 \\ & ② \; \boldsymbol{y} \geq \mathbf{0}. \end{cases}$$

Sequence-Form LP

- Can rewrite g(x) as a linear program

$$g(\boldsymbol{x}) = \begin{cases} \min & (\mathbf{A}^\top \boldsymbol{x})^\top \boldsymbol{y} \\ \text{s.t.} & ① \; \mathbf{F}_2 \boldsymbol{y} = \boldsymbol{f}_2 \\ & ② \; \boldsymbol{y} \geq \mathbf{0}. \end{cases}$$

- By LP duality, we can equivalently write

$$g(\boldsymbol{x}) = \begin{cases} \max & \boldsymbol{f}_2^\top \boldsymbol{v} \\ \text{s.t.} & ① \; \mathbf{F}_2^\top \boldsymbol{v} \leq \mathbf{A}^\top \boldsymbol{x} \\ & ② \; \boldsymbol{v} \text{ free.} \end{cases}$$

Sequence-Form LP

- Now we just plug this back into the original saddle-point problem
  - Just need to add sequence form constraint for x
- The solution to this LP is a Nash equilibrium strategy for player 1
  - Can similarly find NE for player 2

$$\begin{aligned}
\max \quad & \boldsymbol{f}_2^\top \boldsymbol{v} \\
\text{s.t.} \quad & ① \ \mathbf{A}^\top \boldsymbol{x} - \mathbf{F}_2^\top \boldsymbol{v} \geq \mathbf{0} \\
& ② \ \mathbf{F}_1 \boldsymbol{x} = \boldsymbol{f}_1 \\
& ③ \ \boldsymbol{x} \geq \mathbf{0}, \ \ \boldsymbol{v} \ \text{free.}
\end{aligned}$$