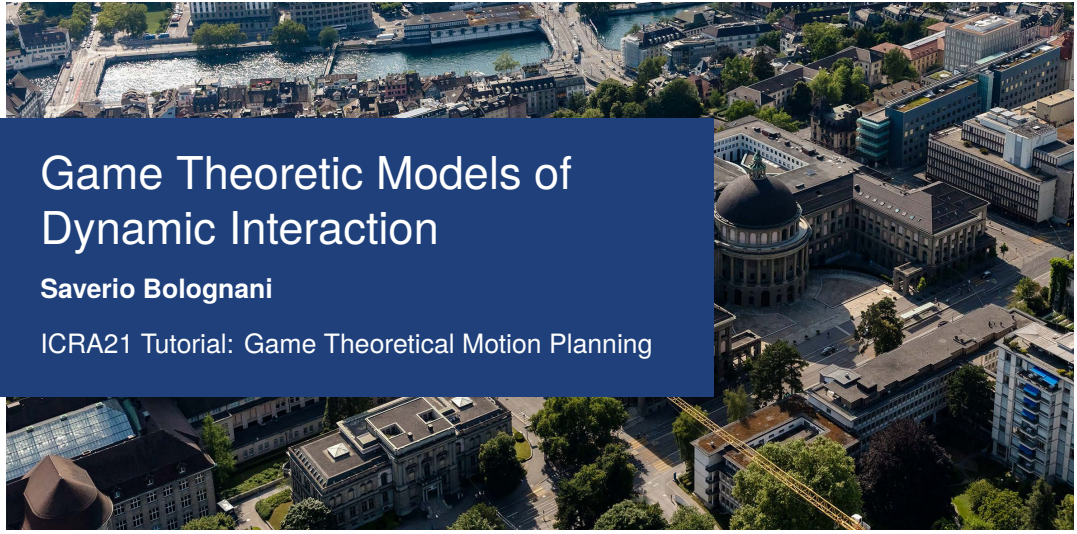
ETH zürich

AUTOMATIC CONTROL LABORATORY



Game Theoretic Models of Dynamic Interaction

Saverio Bolognani

ICRA21 Tutorial: Game Theoretical Motion Planning

Outline

1. Basic concepts and definitions

2. Special games and How to solve them
   - Zero-sum games
   - Potential games
   - Congestion games

3. Multi-stage games
   - Extensive form
   - Feedback games
   - Backward induction
   - From multi-stage to loop-models
   - Closed-loop and Open-loop Nash Equilibria

Outline

1. Basic concepts and definitions

2. Special games and How to solve them
   - Zero-sum games
   - Potential games
   - Congestion games

3. Multi-stage games
   - Extensive form
   - Feedback games
   - Backward induction
   - From multi-stage to loop-models
   - Closed-loop and Open-loop Nash Equilibria

Elements of a Game

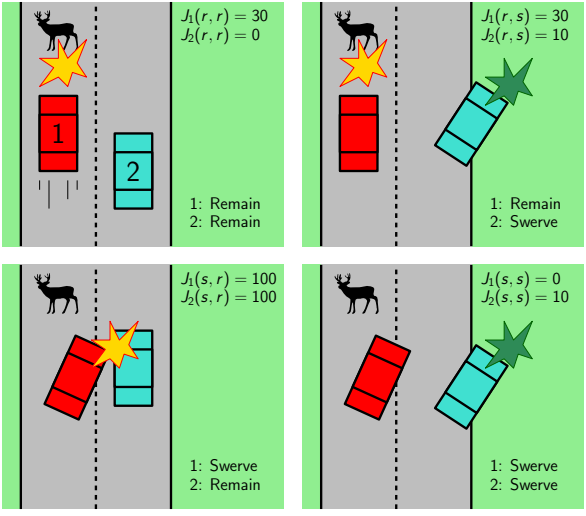
Definition

Game theory is the study of mathematical models of conflict and cooperation between rational decision-makers.

Several things are needed to **characterize a game**:

- The **players** are the agents that make decisions
- The **actions** available to each player at each decision point
- The **information structure** specifies what each player knows before making each decision, in particular with respect to other players' decisions
- The **outcome** for each player, which depends on all players' decisions

Example: Joint maneuver



Matrix representation

For games with

- two players P_1 and P_2
- a finite number of actions $\quad \Gamma = \{\gamma_1, \dots, \gamma_n\} \quad \Sigma = \{\sigma_1, \dots, \sigma_m\}$
- outcomes $J_1(\gamma, \sigma)$ and $J_2(\gamma, \sigma)$

we often adopt the compact representation

$$\begin{array}{c} \\ \text{Remain} \\ \text{Swerve} \end{array} \begin{array}{c} \begin{array}{cc} \text{Remain} & \text{Swerve} \end{array} \\ \begin{bmatrix} (30, 0) & (30, 10) \\ (100, 100) & (0, 10) \end{bmatrix} \end{array} \quad \text{or} \quad A = \begin{bmatrix} 30 & 30 \\ 100 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 10 \\ 100 & 10 \end{bmatrix}$$

where each element of the matrix corresponds to the resulting outcome for the two players

$$a_{ij} = J_1(\gamma_i, \sigma_j), \qquad b_{ij} = J_2(\gamma_i, \sigma_j)$$

“Solving a game”

Predict / compute / mandate a pair of strategies (γ^*, σ^*) such that

- no player can do better by **unilaterally changing their decision**
  (It doesn't say anything about other strategy pairs)
- both players show **no regret** after observing the outcome
- if players **repeat** the game after playing (γ^*, σ^*), none of them is going to change decision.
  (Repetition of the game is not necessarily happening).

The key definition that allows to formalize these ideas is the concept of **best response**.

Best response

$$R_1(\sigma) = \arg\min_{\gamma \in \Gamma} J_1(\gamma, \sigma) \qquad R_2(\gamma) = \arg\min_{\sigma \in \Sigma} J_2(\gamma, \sigma)$$

Nash equilibrium

Nash Equilibrium

Given a static two-player game described by the two payoff functions $J_1(\gamma, \sigma)$ and $J_2(\gamma, \sigma)$, (for P_1 and P_2, respectively), we say that the pair of actions $\gamma^* \in \Gamma$ and $\sigma^* \in \Sigma$ are a **Nash Equilibrium** if

$$\gamma^* \in R_1(\sigma^*) = \arg\min_{\gamma \in \Gamma} J_1(\gamma, \sigma^*)$$

and

$$\sigma^* \in R_2(\gamma^*) = \arg\min_{\sigma \in \Sigma} J_2(\sigma, \gamma^*)$$

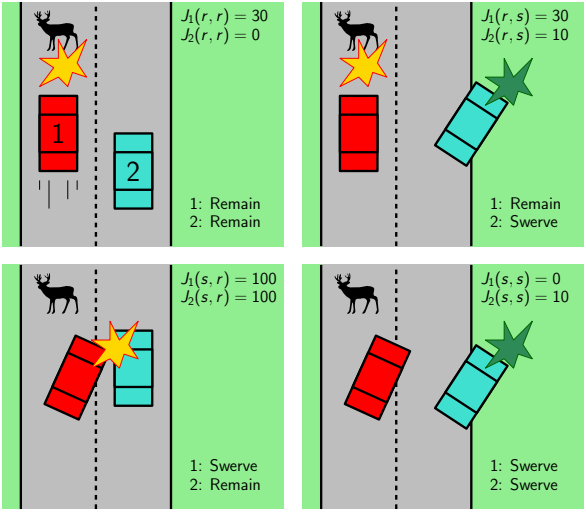
When the payoff is expressed via the matrices A and B, $\gamma_{i^*} \in \Gamma$ and $\sigma_{j^*} \in \Sigma$ are a Nash Equilibrium if

$$a_{i^*j^*} \leq a_{kj^*} \quad \forall k = 1, \ldots, m$$

and

$$b_{i^*j^*} \leq b_{i^*k} \quad \forall k = 1, \ldots, n.$$

Example: Joint maneuver



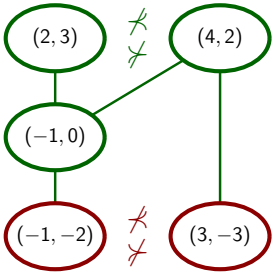
Example: Joint maneuver

	Remain	Swerve
Remain	$(30, 0)$	$(30, 10)$
Swerve	$(100, 100)$	$(0, 10)$

- Player 1's best responses: $R_1(\text{remain}) = \text{remain}$, $R_1(\text{swerve}) = \text{swerve}$
- Player 2's best responses: $R_2(\text{remain}) = \text{remain}$, $R_2(\text{swerve}) = \text{swerve}$
- Nash equilibria
  - **(remain,remain)**, outcome $(30, 0)$
  - **(swerve,swerve)**, outcome $(0, 10)$
- The two Nash equilbria are not **interchangeable** (i.e., **(remain, swerve)** and **(swerve,remain)**) are not Nash equilibria.

Which strategy will/should the two cars play?

Multiple Nash Equilibria



Poset: Partially ordered set

$$(-1,0) \prec (1,2) \qquad \text{but} \qquad \begin{matrix}(1,0) \not\prec (-1,2)\\ (1,0) \not\succ (-1,2)\end{matrix}$$

Hasse diagram: *upward edge* means that $x \prec y$ and there is no z such that $x \prec z \prec y$

Minimal element: an element of the poset that is not greater than any other element

Admissible Nash Equilibria

A Nash equilibrium (γ^*, σ^*) is **admissible** if there is no other Nash equilibrium $(\tilde{\gamma}, \tilde{\sigma})$ such that

$$J_1(\tilde{\gamma}, \tilde{\sigma}) \leq J_1(\gamma^*, \sigma^*) \quad \textbf{and} \quad J_2(\tilde{\gamma}, \tilde{\sigma}) \leq J_2(\gamma^*, \sigma^*)$$

with at least one of the two inequalities **strict**.

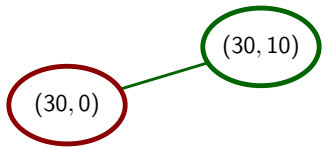
Example: Joint maneuver

Very hard to predict!

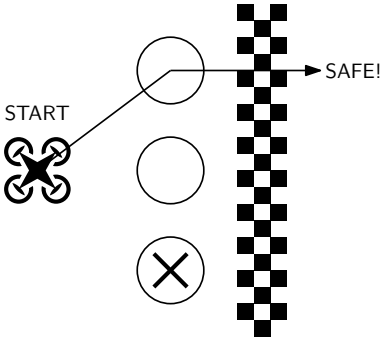
- **Multiple admissible Nash equilibria** which
  - have **different values**
  - are **not interchangeable**



- Conservative behavior: both Players minimize their **worst-case outcome (security strategy)**
  - **(remain, swerve)**, with outcome $(30, 10)$ (worse than both NE...)
- **Mechanism design**, i.e. change the game to induce a unique admissible Nash equilibrium
  - example: a **fine** of 30 if you change lane and hit another car



Example: Escape game



- Player 1 wins if it reaches the finish line safely
- Player 2 can block one of the three ways to safety

	block LEFT	block MIDDLE	block RIGHT
go LEFT	$(+1, -1)$	$(-1, +1)$	$(-1, +1)$
go MIDDLE	$(-1, +1)$	$(+1, -1)$	$(-1, +1)$
go RIGHT	$(-1, +1)$	$(-1, +1)$	$(+1, -1)$

No Nash Equilibrium exists

- $R_2(\text{go LEFT}) = \{\text{block LEFT}\}$
- $R_1(\text{block LEFT}) = \{\text{go MIDDLE}, \text{block RIGHT}\}$

Randomized play

- A **mixed strategy** is a stochastic strategy
- Mixed strategies are described by probabilities associated to each action

$$\Gamma = \{\gamma_1, \dots, \gamma_n\} \quad \rightarrow \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \Sigma = \{\sigma_1, \dots, \sigma_m\} \quad \rightarrow \quad z = \begin{bmatrix} z_1 \\ \vdots \\ z_m \end{bmatrix}$$

$$y \in \mathcal{Y} = \left\{ y \mid \sum_i y_i = 1, y_i \geq 0 \right\} \qquad z \in \mathcal{Z} = \left\{ z \mid \sum_i z_i = 1, z_i \geq 0 \right\}$$

- **Expected outcome** of a game

$$J_1(y,z) = \sum_{i=1}^{m} \sum_{j=1}^{n} y_i z_j a_{ij} = y^\top A z \qquad J_2(y,z) = \sum_{i=1}^{m} \sum_{j=1}^{n} y_i z_j b_{ij} = y^\top B z$$

- Deterministic strategies are referred to as **pure strategies**

Mixed Nash Equilibrium

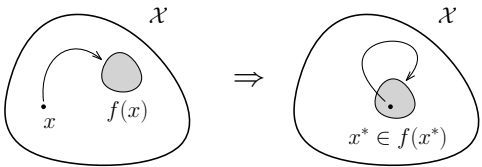
Mixed Nash Equilibrium

Given a static two-player game described by the two payoff matrices A and B (for P_1 and P_2, respectively), we say that the pair of mixed strategies $y^* \in \mathcal{Y}$ and $z^* \in \mathcal{Z}$ are a **mixed Nash Equilibrium** if

$$J_1(y^*, z^*) \leq J_1(y, z^*) \quad \forall y \in \mathcal{Y}$$

$$J_2(y^*, z^*) \leq J_2(y^*, z) \quad \forall z \in \mathcal{Z}$$

Nash Existence Theorem: a mixed Nash Equilibrium is always guaranteed to exist.



Key proof step:
Kakutani's Fixed Point theorem applied to

$$(y, z) \mapsto R(y, z) = (R_1(z), R_2(y))$$

Proof

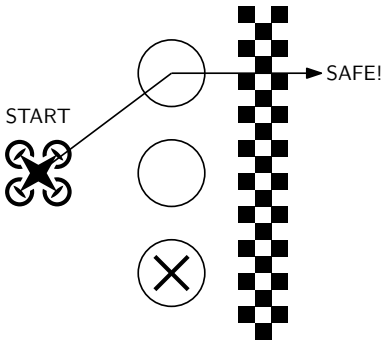
Outline

1. Basic concepts and definitions

2. Special games and How to solve them
   - Zero-sum games
   - Potential games
   - Congestion games

3. Multi-stage games
   - Extensive form
   - Feedback games
   - Backward induction
   - From multi-stage to loop-models
   - Closed-loop and Open-loop Nash Equilibria

Zero-sum games



	block LEFT	block MIDDLE	block RIGHT
go LEFT	$(+1,-1)$	$(-1,+1)$	$(-1,+1)$
go MIDDLE	$(-1,+1)$	$(+1,-1)$	$(-1,+1)$
go RIGHT	$(-1,+1)$	$(-1,+1)$	$(+1,-1)$

$$J(y,z) := J_1(y,z) = -J_2(y,z)$$

Saddle point (Nash Equilibrium in Zero-sum games)

$$J(y^*,z) \leq J(y^*,z^*) \leq J(y,z^*) \quad \forall y \in \mathcal{Y},\ z \in \mathcal{Z}$$

Multiple possible interpretations:

- Adversarial agents
- Agent against nature (worst-case robustness)
- Equivalent non-zero sum games (e.g. security games)

Security policies in zero-sum games

Worst-case robustness: minimize the expected game value as if the other player was allow to respond.

Security level

Mixed security levels are defined as

$$\overline{V} := \min_{y \in \mathcal{Y}} \max_{z \in \mathcal{Z}} J(y, z) \qquad \underline{V} := \max_{z \in \mathcal{Z}} \min_{y \in \mathcal{Y}} J(y, z)$$

Security policy

A **mixed security policy** is any

$$\overline{y} \in \arg\min_{y \in \mathcal{Y}} \max_{z \in \mathcal{Z}} J(y, z) \qquad \underline{z} \in \arg\max_{z \in \mathcal{Z}} \min_{y \in \mathcal{Y}} J(y, z)$$

Saddle Point vs. Security Levels in zero-sum games

Theorem

- *The two* ***security levels*** *coincide*

$$\overline{V} = \min_{y \in \mathcal{Y}} \max_{z \in \mathcal{Z}} J(y, z) = \max_{z \in \mathcal{Z}} \min_{y \in \mathcal{Y}} J(y, z) = \underline{V}$$

- *The security policies are* ***saddle-point (Nash Equilibrium) strategies***

$$J(\overline{y}, z) \leq J(\overline{y}, \underline{z}) \leq J(y, \underline{z}) \quad \forall y \in \mathcal{Y},\ z \in \mathcal{Z}$$

Proof

The fundamental reason why zero-sum games are important:

- Nash equilibria are easy to compute (**linear programming**)
- All Nash equilibria have the same value ($\rightarrow$ **value of the game**)
- All Nash equilibria are **admissible** and **interchangeable**

Computing Nash equilibria in zero-sum games

By using the linearity of the expected outcome $J(y,z) = \sum_i \sum_j y_i z_j a_{ij}$ in y and z, we have that

$$\overline{V} := \min_{y \in \mathcal{Y}} \underbrace{\max_{z \in \mathcal{Z}} \sum_{i=1}^{m} \sum_{j=1}^{n} y_i z_j a_{ij}}_{\text{linear program}} = \min_{y \in \mathcal{Y}} \underbrace{\max_{j} \left(\sum_{i=1}^{m} y_i a_{ij} \right)}_{\text{basic feasible solution}}$$

Computational method – Linear programming for zero-sum games

Player 1's Nash equilibrium strategy y and the value V of a zero-sum game can be found by solving

$$\begin{aligned} \min_{y,V} \quad & V \\ \text{subject to} \quad & \sum_{i=1}^{m} y_i a_{ij} \leq V, \quad j = 1, \ldots, n \\ & y \in \mathcal{Y},\ V \in \mathbb{R} \end{aligned}$$

Potential games

Consider a game with N **players** and let

- γ_i be the strategy of Player i
- γ_{-i} be the strategies of the other players

Best reponse

$$R_i(\gamma_{-i}) = \arg\min_{\gamma_i \in \Gamma_i} J_i(\gamma_i, \gamma_{-i})$$

Generalized NE $\gamma_i \in \Gamma_i(\gamma_{-i})$

Definition

A game is a **potential game** if there exists a **potential function** $P : \gamma_1 \times \gamma_2 \times \ldots \times \gamma_N \to \mathbb{R}$ such that for every player i

$$J_i(\gamma_i', \gamma_{-i}) - J_i(\gamma_i'', \gamma_{-i}) = P(\gamma_i', \gamma_{-i}) - P(\gamma_i'', \gamma_{-i}) \qquad \forall \gamma_i', \gamma_i'', \gamma_{-i}$$

Computational method – Centralized non-linear optimizaton for potential games

In potential games, **pure Nash equilibria** correspond to **directional minima** of the potential.

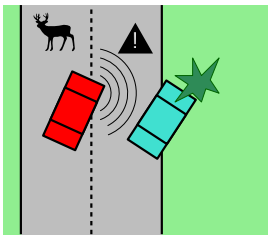
Best response dynamics

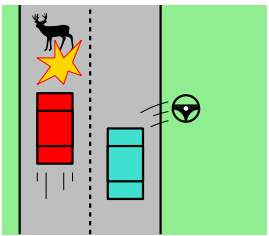
Proposition

In **finite potential games**, **iterated best response** converges to a pure Nash equilibrium.

Example of iterated best response

	Remain	Swerve
Remain	$(30, 0)$	$(30, 10)$
Swerve	$(100, 100)$	$(0, 10)$





Iterated best response

Computational method – Iterated best response

Consider an initial pure strategy profile $\gamma^0 = (\gamma_1^0, \gamma_2^0, \ldots, \gamma_n^0)$. Let $k = 0$.

1. If γ^k is a pure Nash equilibrium $\rightarrow$ **stop**
2. Else, there exists a player i for which $\gamma_i^k \notin R(\gamma_{-i}^k)$.
3. Update: $\gamma_i^{k+1} \in R(\gamma_{-i}^k), \qquad \gamma_{-i}^{k+1} = \gamma_{-i}^k$
4. $k = k + 1$, goto step 1.

- An **iterative computation method** but also a possible **distributed negotiation protocol**
- It converges to a Nash equilbrium that depends on the **initial conditions**
- It does not converge only to admissible Nash equilibria
- A modified version of the iterated best response is available for
  - games with **continuous action spaces**
  - **Generalized Nash Equilibria problems**

Regularized IBR, ϵ-IBR

Congestion games

- Set $\mathcal{M} = \{1, \dots, M\}$ of **resources**.
- Each strategy corresponds to **using a subset of the resources**

  $\gamma_i \subseteq \mathcal{M}$ for example $\gamma_i^{(1)} = \{4\}$, $\gamma_i^{(2)} = \{2, 4, 6\}$, $\dots$

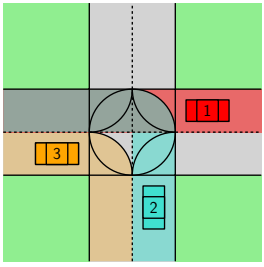
- The **load** on resource j is the number of players who use it

$$\ell_j(\gamma) := |\{i \mid j \in \gamma_i\}|$$

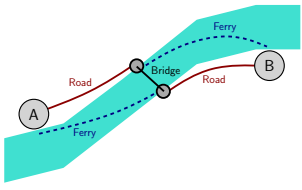
- The **cost** for each player is a non-decreasing function of the load on the resources they are using

$$J_i(\gamma) = \sum_{j \in \gamma_i} f_j(\ell_j(\gamma))$$

Sharing the road space



Sharing the road network



Congestion games are potential games

Theorem

The function

$$P(\gamma) = \sum_{j=1}^{M} \sum_{k=1}^{\ell_j(\gamma)} f_j(k).$$

is a potential function for the corresponding congestion game. **Proof**

Examples: data networks, wireless communication, traffic congestion, manifacturing, power grids, ...

Except for special cases, minimizing the potential function does not correspond to minimizing the **social cost** in a congestion game

$$W(\gamma) = \sum_{j=1}^{M} \ell_j(\gamma) f_j(\ell_j(\gamma)).$$

Braess Paradox, Price of Anarchy

TECHNOLOGY

The Perfect Selfishness of Mapping Apps

Apps like Waze, Google Maps, and Apple Maps may make traffic conditions worse in some areas, new research suggests.



Why Some Cities Have Had Enough of Waze





We are not just facing data security and privacy issues caused by the mass digitalization of our ordinary lives, we are also facing physical danger and lifestyle degradation in areas we would never had imagined that technology would affect negatively. Photo: HT/File

Of Map Apps and Traffic Jams

For the Good of Society — and Traffic! — Delete Your Map App

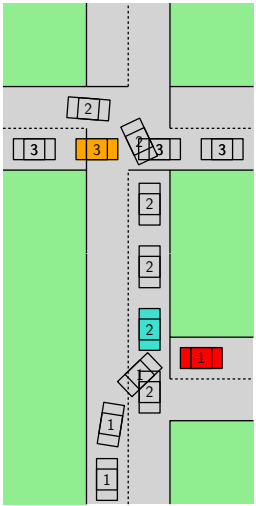
Outline

1. Basic concepts and definitions

2. Special games and How to solve them
   - Zero-sum games
   - Potential games
   - Congestion games

3. Multi-stage games
   - Extensive form
   - Feedback games
   - Backward induction
   - From multi-stage to loop-models
   - Closed-loop and Open-loop Nash Equilibria

Multistage games

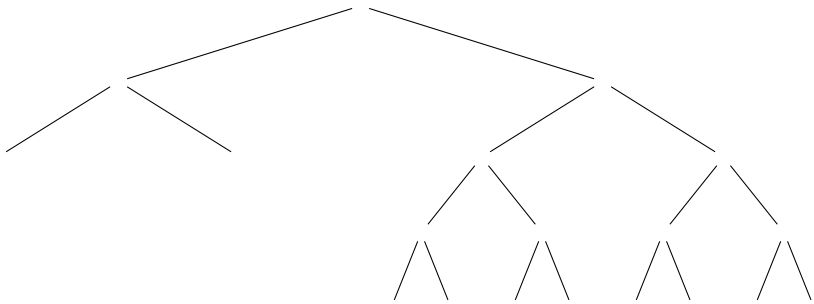


In practical dynamic interactions, we are often interested in **more complicated decision problems**:

- multiple stages
- sequential decisions rather than simultaneous
- variable number of stages
- partial information availability
- memory-constrained players

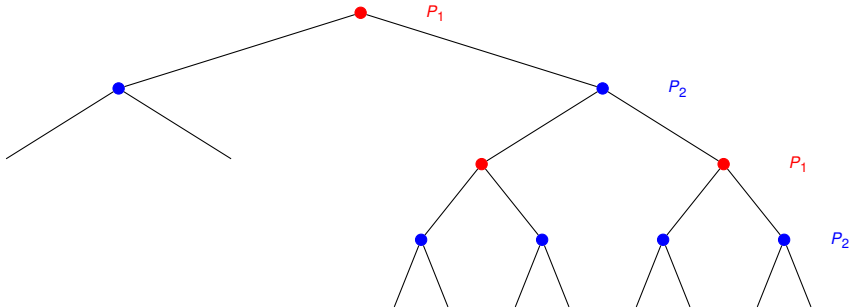
$\rightarrow$ **multi-stage games**

Extensive form: tree representation



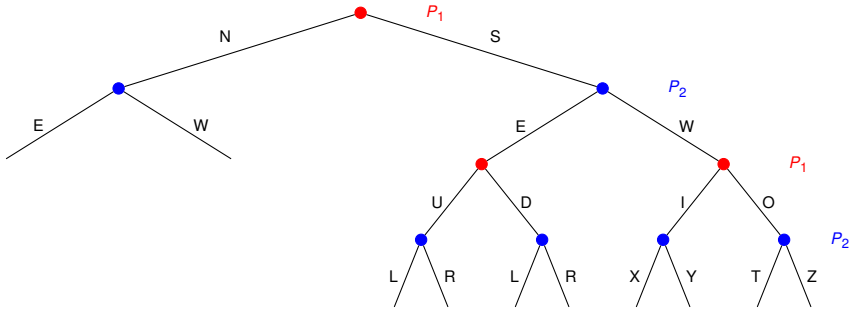
- The game evolves from the **root** to the **leaves**

Extensive form: tree representation



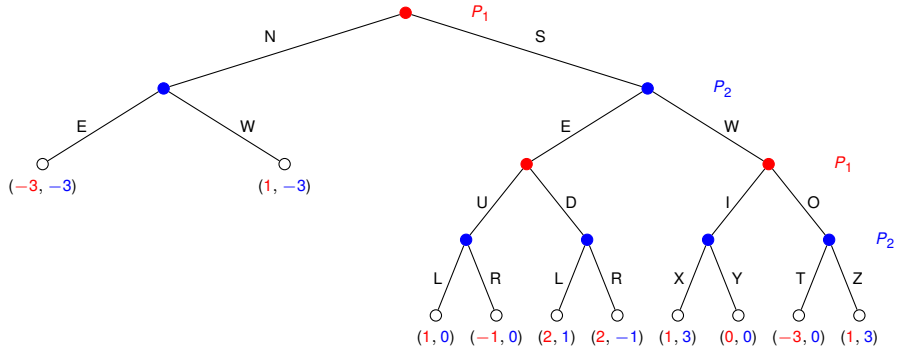
- Each **node** of the tree corresponds to a player's turn to take a decision

Extensive form: tree representation



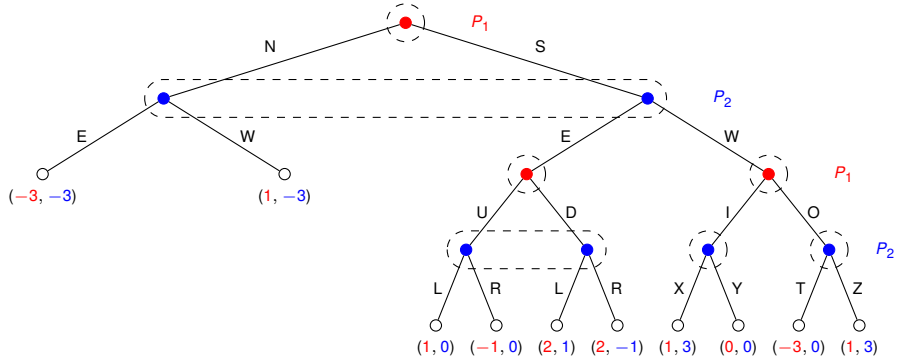
- **Links** correspond to **actions**

Extensive form: tree representation



- Each **leaf** is associated to a final **outcome**

Extensive form: tree representation

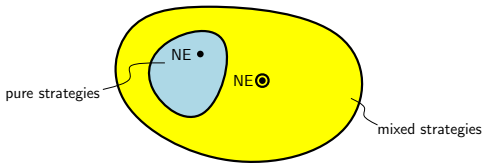


- Nodes of each player are divided into **information sets** (ambiguity of the player's knowledge)

Strategies in multi-stage games

Definitions

- **Pure strategy**
  A map that assigns an action to each information set
  $$\mathcal{I}_h \mapsto u_h = \gamma(\mathcal{I}_h) \in \mathcal{U}_h,$$
- **Mixed strategy**
  A **probability distribution** over the pure strategies $\gamma_i \in \Gamma$
  $$y \in \mathcal{Y} \subset \mathbb{R}^m, \quad m = |\Gamma|, \qquad y_i = \mathbb{P}(\gamma_i)$$

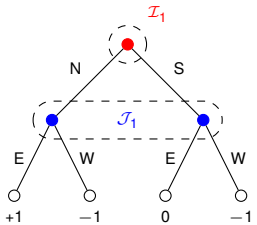


Under these definitions, all the previous results hold.

- Existence of Nash Equilibria
- Computational methods
- Properties of special classes of games

From extensive form to matrix form

Simultaneous play

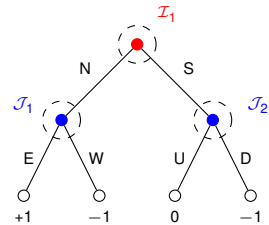


IS : $\{\mathcal{I}_1\}$ IS : $\{\mathcal{J}_1\}$

$\mathcal{U}_1 = \{N, S\}$ $\mathcal{V}_1 = \{E, W\}$

	$\sigma(\mathcal{J}_1)$=E	$\sigma(\mathcal{J}_1)$=W
$\gamma(\mathcal{I}_1)$=N	+1	−1
$\gamma(\mathcal{I}_1)$=S	0	−1

Sequential play



IS : $\{\mathcal{I}_1\}$ IS : $\{\mathcal{J}_1, \mathcal{J}_2\}$

$\mathcal{U}_1 = \{N, S\}$ $\mathcal{V}_1 = \{E, W\},\ \mathcal{V}_2 = \{U, D\}$

	$\sigma(\mathcal{J}_1)$=E $\sigma(\mathcal{J}_2)$=U	$\sigma(\mathcal{J}_1)$=E $\sigma(\mathcal{J}_2)$=D	$\sigma(\mathcal{J}_1)$=W $\sigma(\mathcal{J}_2)$=U	$\sigma(\mathcal{J}_1)$=W $\sigma(\mathcal{J}_2)$=D
$\gamma(\mathcal{I}_1)$=N	+1	+1	−1	−1
$\gamma(\mathcal{I}_1)$=S	0	−1	0	−1

Solution of the equivalent matrix game

Computational method – Nash Equilibrium search on the equivalent matrix game

Step 1: Convert the multi-stage game into a static game (matrix form)
Step 2: Solve the static game to find the Nash Equilibria

Important drawback

- Problem size which is **exponential** in the number of information sets

Better computational tools exist if we restrict the search to a special class of multi-stage games.

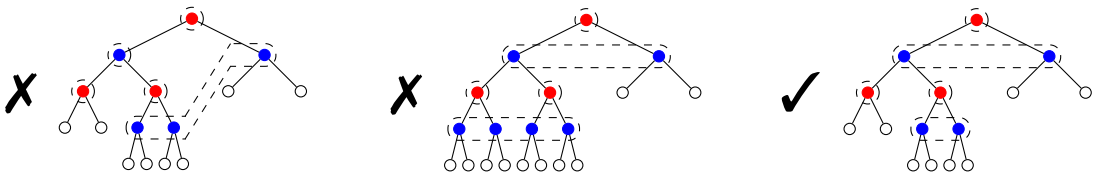
Feedback games

Information sets can be quite exotic, and represent limited player memory and awareness.

We restrict ourselves to games in which all players have **perfect recall** of the past.

Feedback games

1. no information set spans over multiple stages
2. the subtrees in each stage are not “connected” by any information set

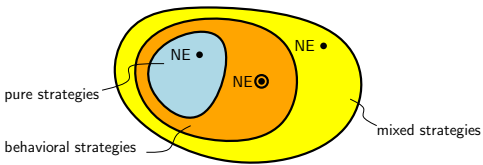


Behavioral strategies

Definition

Behavioral strategy
A map that assigns a **probability distribution** over the actions to each information set

$$\mathcal{I}_h \mapsto \gamma^b(\mathcal{I}_h) \in \mathcal{Y}_h \subset \mathbb{R}^{|\mathcal{U}_h|}$$



In **feedback games** it is **not restrictive** to consider only behavioral strategies.

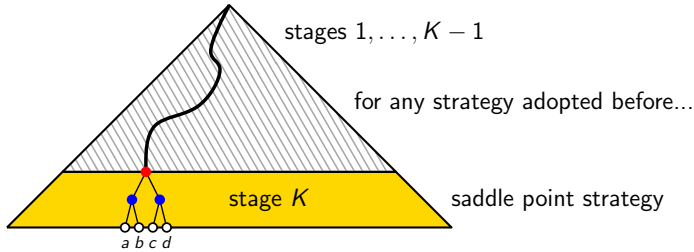
Kuhn's theorem

Computational method: backward induction

1. The set of behavioral strategies is **computationally tractable**
   - Backward induction allows to find NE in behavioral strategies
   - The low search space dimension makes the algorithm efficient
2. Backward induction returns a **subgame perfect** strategy.

Backward induction

Apply the **single stage-procedure** to the last stage of the game.



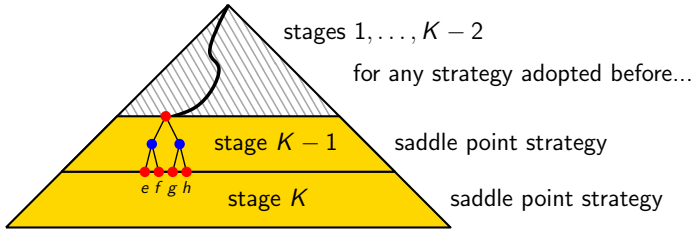
We can do that because its a feedback game: *each subtree is separate from other each other*.

We obtain

- a pair $(\gamma^{(K)*}, \sigma^{(K)*})$ of **NE behavioral strategies** for every subgame at stage K
- the resulting **value** for each of these single stage games.

Backward induction

Label the roots of the stage-K subgames with their value, and proceed towards the root.

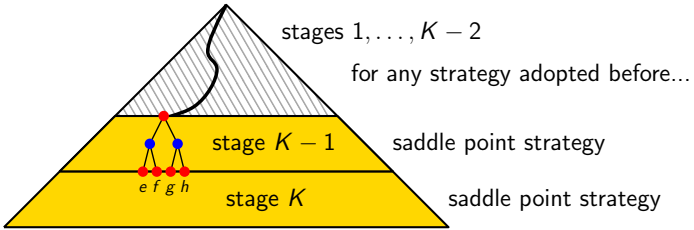


Assuming that NE strategies will be played at stage K, we obtain

- a pair $(\gamma^{(K-1)*}, \sigma^{(K-1)*})$ of **NE behavioral strategies** for stage $K-1$
- the **value** of each one of these single stage games.

Backward induction

Label the roots of the stage-K subgames with their value, and proceed towards the root.



Subgame perfect Nash Equilibrium

The resulting Nash Equilibrium is a Nash Equilibrium for each subgame.
In other words, it describe a Nash Equilibrium behavior at every stage

- regardless of what happened in the **past** (**robustness**)
- assuming that the Nash Equilibrium strategy is played in the **future**

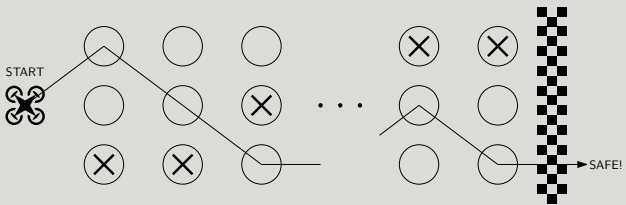
Taming the complexity of multi-stage games

Example

Multi-stage game with

- finite actions (3)
- large number of stages (20)

$\sim 1+9+81+\ldots+3^{20}$ linear programs!

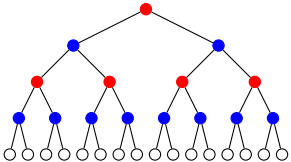


Key idea:

- In every "position", the probability of escaping (and the optimal strategy) do not depend on the past decisions.
- We can perform backward induction on the "positions" (**states!**) rather than on the game tree.

3×20 linear programs instead of 5×10^9

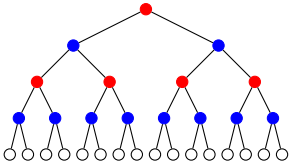
From tree model from loop model



Consider a **feedback game** in extensive form in which at each stage k

- Player 1 action: $u_k \in \mathcal{U}_k \subseteq \mathbb{R}^m$
- Player 2 action: $v_k \in \mathcal{V}_k \subseteq \mathbb{R}^n$

From tree model from loop model



Consider a **feedback game** in extensive form in which at each stage k

- Player 1 action: $u_k \in \mathcal{U}_k \subseteq \mathbb{R}^m$
- Player 2 action: $v_k \in \mathcal{V}_k \subseteq \mathbb{R}^n$

Assumption 1

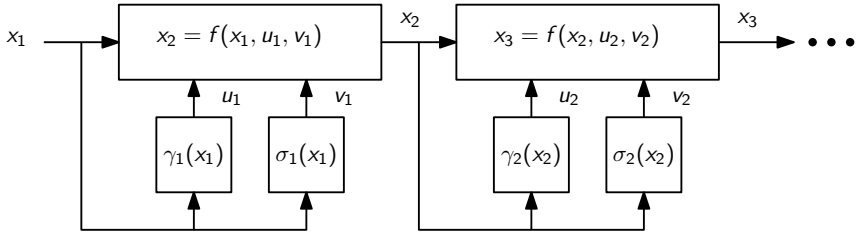
There exists a **state** that evolves at each stage: $x_{k+1} = f(x_k, u_k, v_k)$

Assumption 2

Stage-additive outcome $\sum_{k=1}^{K} g_k(x_k, u_k, v_k)$

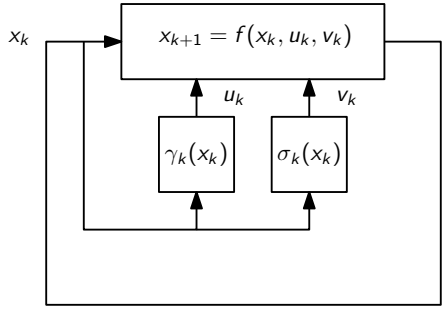
$\Rightarrow$ **At each stage, the state is a sufficient representation of the game for all purposes.**

Loop model



- A **multi-stage game** become a **discrete-time system**
- **Strategies** (maps from information sets to actions) become **feedback laws** (maps from states to inputs)
- Players have the same information available (**simultaneous play**), although variations are possible (output feedback, delays, ...)

Loop model



- A **multi-stage game** become a **discrete-time system**
- **Strategies** (maps from information sets to actions) become **feedback laws** (maps from states to inputs)
- Players have the same information available (**simultaneous play**), although variations are possible (output feedback, delays, ...)

Backward induction in dynamic games

Computational method – Backward induction on the state

Find a **subgame-perfect NE** by iterating backward from the terminal state.
Initialize $V^{(1)}_{K+1}(x_K) = V^{(2)}_{K+1}(x_K) = 0$.

1. **Stage K NE:** Determine γ^*_K, σ^*_K (functions of x_K) that satisfy the Nash-equilibrium condition

$$\gamma^*_K(x_K) = \underset{u}{\operatorname{argmin}}\, g_K(x_K, u, \sigma^*_K(x_K)) + V^{(1)}_{K+1}(f(x_K, u, \sigma^*_K(x_K)))$$

$$\sigma^*_K(x_K) = \underset{v}{\operatorname{argmin}}\, g_K(x_K, \gamma^*_K(x_K), v) + V^{(2)}_{K+1}(f(x_K, \gamma^*_K(x_K), v))$$

2. **Stage K value function:** Let $V^{(1)}_K(x_K)$, $V^{(2)}_K(x_K)$ be the value of the subgame rooted in x_K for the two players.
3. If $K > 1$, then $K \leftarrow K - 1$, **go to step 1**.

$\sim$ Dynamic Programming / Optimal control with a NE condition instead of the Bellman equation

Linear-quadratic case

Finding the **subgame-perfect NE strategy** for a dynamic game is a very complicated task for all dynamic games but a very few exceptions.

Multiagent LQR

- **Linear** state update *f*
  Example: kinematic equations
- **Quadratic** cost function *g*
  Example: speed and position tracking

2-player LQR



Open-loop NE strategy

More often, a numerical solution is available for the **open-loop** joint problem

$$
\begin{aligned}
u_0^*, u_1^*, \ldots, u_{K-1}^* \quad = \quad & \underset{u_0,\ldots,u_{K-1}}{\operatorname{argmin}} \quad \sum_{k=0}^{K-1} g_k(x_k, u_k, v_k^*) \\
& \text{subject to} \quad x_{k+1} = f_k(x_k, u_k, v_k^*) \quad \text{for } k = 0, \ldots, K-1 \\
v_0^*, v_1^*, \ldots, v_{K-1}^* \quad = \quad & \underset{v_0,\ldots,v_{K-1}}{\operatorname{argmin}} \quad \sum_{k=0}^{K-1} g_k(x_k, u_k^*, v_k) \\
& \text{subject to} \quad x_{k+1} = f_k(x_k, u_k^*, v_k) \quad \text{for } k = 0, \ldots, K-1
\end{aligned}
$$

Computational method – Joint solution of coupled optimal control problems

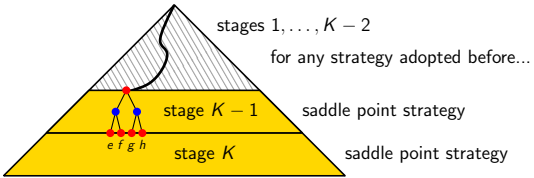
Usually via local optimality conditions such as **KKT conditions** or **variational inequalities (VI)**.

Open-loop vs Closed-loop

Closed-loop NE

State feedback
Subgame-perfect strategy

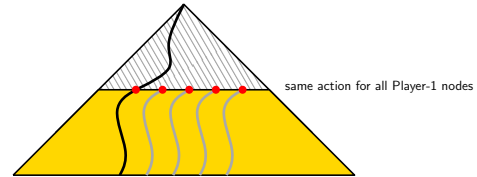
$u_0^*(x), u_1^*(x), \dots u_{K-1}^*(x)$



Open-loop NE

Planned trajectory
Non-subgame-perfect strategy

$u_0^*, u_1^*, \dots, u_{K-1}^* \in \mathbb{R}^m$

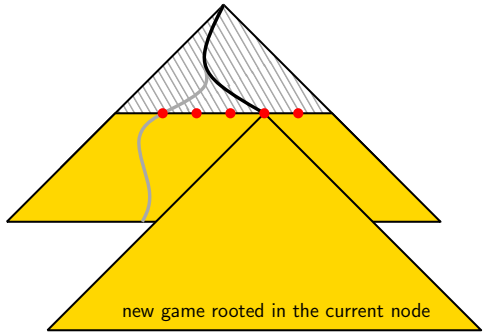


Receding horizon

Computational method – Receding-horizon open-loop Nash Equilibrium

Re-calculate a NE strategy from the new starting point (i.e., for the subtree rooted in the current node).

- Computationally not as efficient as closed-loop **backward induction** (when possible!)
- Optimization has to be solved **in real time**



Outline

1. Basic concepts and definitions

2. Special games and How to solve them
   - Zero-sum games
   - Potential games
   - Congestion games

3. Multi-stage games
   - Extensive form
   - Feedback games
   - Backward induction
   - From multi-stage to loop-models
   - Closed-loop and Open-loop Nash Equilibria

ETH zürich



Saverio Bolognani
bsaverio@ethz.ch