Course Overview: AAMAS 2025

Outline

- **Course organization**
  - Instructors
  - Learning objectives
  - Bibliography
  - Grading
  - Project
  - Schedule
  - Office hours
- **Course planning**

Instructors: Campus Alameda

Manuel Cabido Lopes Responsável
manuel.lopes@tecnico.ulisboa.pt

Ana Vilaça Carrasco

Miguel Afonso Tomás Faria
miguel.faria@tecnico.ulisboa.pt

Rui Filipe Fernandes Prada Responsável
rui.prada@tecnico.ulisboa.pt

Diogo Carvalho

Pedro Pinto Santos

Instructors: Campus Oeiras (Taguspark)

Manuel Cabido Lopes Responsável
manuel.lopes@tecnico.ulisboa.pt

Jacopo Silvestrin

Rui Filipe Fernandes Prada Responsável
rui.prada@tecnico.ulisboa.pt

Learning Objectives

- To introduce key concepts, different views, and applications of **agent systems**
- To present a structured view on **agent architectures**
  - Perception, decision making, and action
- **To map real-world problems into agent-oriented tasks**
  - **Design reactive**, **deliberative**, and **hybrid agents**
  - **Develop agents with decision-making algorithms**
  - **Design societies of agents** to solve specific problems

Bibliography

Primary

- ***Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations,*** 1st Ed, Yoav Shoham and Kevin Leyton-Brown, Cambridge University Press, 2009 **[S+LB09]**

- ***An Introduction to MultiAgent Systems***, 2nd Ed, Michael Wooldridge, Wiley & Sons Ltd, 2009 **[W09]**

Bibliography

Secondary

- ***A Concise Introduction to Multiagent Systems and Distributed Artificial Intelligence***, Nikos Vlassis, Morgan & Claypool, 2007 **[V07]**

- ***Primer In Game Theory***, Robert Gibbons, Pearson Education Limited, 1992

Grading

- **Exam** **50%** (minimum **8v**)
  Date: Wednesday, **25/06/2025**
- **Exam Retrial** (minimum **8v**)
  Date: Tuesday, **08/07/2025**

- **Project** **50%** (minimum **8v**)
  Proposal: develop an agent system to answer a complex problem, reimplement a system
  Proposal due date: Friday, **09/05/2025**
  Project due date: Friday, **06/06/2025**
  Project presentation (in the lab): **week of 09/06/2025**

Evaluation (including deadlines) is the same
for working students: no exceptions!

Deadlines are fixed: no changes are allowed

Project and Tests – Honor Code

- Cheating is not tolerated in this course!
- Projects are to be developed by each group individually
- Exams are to be done by each student individually
- **Any cheating will lead to an immediate fail of all students involved**

Schedule

	Seg 5/5	Ter 5/6	Qua 5/7	Qui 5/8	Sex 5/9
07:00					
08:00					
09:00			08:30 - 10:30 T A5		09:00 - 12:00 L V1.12
10:00		10:30 - 12:30 T A5	10:30 - 13:30 L 0 - 16	10:00 - 12:0 T GA5 10:30 - 13: L 0 - 16	
11:00					
12:00					12:30 - 15:30 L V0.08
13:00					
14:00		14:00 - 17:00 L V1.24			
15:00	15:00 - 17:00 T GA1		15:30 - 18:30 L V0.06		
16:00					
17:00					
18:00					

Office Hours

- Check the office hours in the course webpage (at Fenix)
- Questions about lectures, projects, lab assignments, etc. → **Office hours**
- Questions about logistics → **e-mail**

Planning

Week	Lectures	Book Chapters	Lab	Dates
28 April	1. Introduction **Holiday (Alameda)**	Chapter 2 [W09] Chapters 3, 4 & 5 [W09]	(Registration)	
5 May	2. Agent Architectures 3. Rational Agents + Normal-form Games	Chapters 3, 4 & 5 [W09] Chapter 3 [S+LB09]	Proposal / *Reactive Agents*	Proposal due date
12 May	4. Normal-form Games II and Mixed Strategies 5. Coordination Games I	Chapter 3 [S+LB09] Chapter 3 [V07] Chapter 2 [S+LB09] Chapter 4 [V07]	Proposal Presentation	
19 May Dia do Técnico	6. Extensive-form Games I 7. Extensive-form Games II	Chapter 5 [S+LB09]	*Coordination*	
26 May	8. Repeated Games (No tagus começa na semana anterior) 9. Multiagent Learning I	Chapters 6, 7, 11 [S+LB09]	*Learning Agents*	
2 June	10. Multiagent Learning II 11. Bayesian Games and Auctions	Chapters 6, 7, 11 [S+LB09]	Project	Project due date
9 June	**Holiday (Tagus)** 12. Bayesian Games and Auctions	Chapter 6 [S+LB09]	Project Presentation	

Project

- Implement an agent system
- Run empirical studies
- Deliver
  - The system source code and executable
  - A report about the agents' system and the empirical evaluation
  - A video demonstrating the agents in action
  - 15 minute presentation in the lab

Project

Reimplement an existing system, choose a research paper to reproduce

or

Identify a relevant real-world problem or social context to study

- Specify the problem using a (multi)agent system
  - AA: decision under uncertainty, complex environments, learning
  - MAS: cooperation, negotiation, multiagent learning

- Specify and develop the agent(s) intelligence using reactive, deliberative and adaptive behavior
- Run comprehensive simulations and present empirical results and discussion of agent(s) behavior

Thank You

rui.prada@tecnico.ulisboa.pt

TÉCNICO LISBOA

Properties of agent systems

Perspectives on Agents and Environments

Outline

- **What is an agent?**
- Intelligent agents and agent properties
- Environment properties
- Agent applications

What is an Agent?

- Remember *Autonomy* is a key point about agents
  - In other words, they are capable of *acting independently*, exhibiting *control over their internal state*

- Thus: *an* agent *is a computer system capable of* autonomous action *in some environment in order to meet its* design objectives

What is an Agent?

- We think of an agent as being in a closed-couple continual interaction with its environment:

sense – decide – act – sense – decide – act…

Outline

- What is an agent?
- **Intelligent agents and agent properties**
- Environment properties
- Agent applications

Simple (Uninteresting) Agents

- Thermostat
  - Goal to *maintain room temperature*
  - Goal delegated by user
  - Actions: heat on / heat off
  - Temperature sensor

Simple (Uninteresting) Agents

- Screen saver
  - Software that *blanks the screen* (or presents moving images) *after a period of user inactivity*
  - Goal avoid screen burning and to prevent other people from viewing desktop contents while the user is away
  - Actions: blank the screen / show desktop

- Why are they uninteresting?
  - They are *trivial* from a *decision-making perspective*

Intelligent Agents

- Intelligent agents typically exhibit the following types of behavior:
  - *reactive*
  - *pro-active*
  - *social*

Reactivity

- What happens if a program's *environment is guaranteed to be fixed*?
  - a program can just execute blindly

- However, the real world is not like that
  - Things change, information is incomplete.
  - Many (most?) interesting environments are *dynamic*

Reactivity

- Software is hard to build for dynamic domains
  - E.g., program must *take into account possibility of failure* – ask itself whether it is worth executing!

- A *reactive* system is one that maintains an *ongoing interaction with its environment*, and *responds to changes* that occur in it (in time for the response to be useful)

Proactiveness

- Reacting to an environment is relatively easy to implement
  - E.g., stimulus → response rules

- But we generally want agents to *do things for us*
  - Hence having *goal directed behavior*

- Pro-activeness = generating and attempting to achieve goals
  - Not driven solely by events from the environment
  - Taking the initiative
  - Recognizing opportunities

Balancing Reactive and Goal-Oriented Behavior

- We want our *agents to be reactive*
  - Thus, responding to changing conditions in an appropriate (timely) fashion
- We want our *agents to systematically work towards long-term goals*
- However, these *two considerations can be at odds with one another*
- Designing an agent that can *balance the two remains an open research problem*

Social Ability

- The real world is a *multi*-agent environment
  - Hence, we cannot go around attempting to achieve goals *without taking others into account*

- Some *goals can only be achieved with the cooperation of others*

Social Ability

- *Social ability* in agents is the ability to interact with other agents (and possibly humans)
  - through *coordination*, *cooperation, negotiation* and *modeling others*

Social Ability

- *Coordination* is managing the interdependencies between activities

- For example:
  - A non-sharable resource in the environment
  - Agents need to coordinate to use this resource

Social Ability

- *Cooperation* is working together as a team to achieve a shared goal

- Typically needed when:
  - No agent can achieve the goal alone
  - Cooperation obtains a better result

Social Ability

- *Negotiation* is the ability to reach agreements on matter of common interest

- For example:
  - You have a TV in your house
  - You want to watch a movie
  - Your roommate wants to watch football
  - Deal: He watches football tonight and you watch a movie tomorrow

Other Agent Properties

- Are there other properties that characterize agents?
  - **Autonomy**
  - Adaptivity
  - Rationality
  - Curiosity
  - Believability
  - Mobility

Agent properties

Autonomy

- agent's *ability to act independently* and thus determine how to achieve its delegated goals/tasks

Adaptivity

- agent's *ability to learn from experience* (to better interact with the particular environment)

Agent properties

Rationality

- agent's *ability to act in a way that maximizes some utility function*

Curiosity

- Agent's *ability to engage creative imaginative or inquisitive reasoning*

Agent properties

Believability

- agent's *ability to create a suspension of disbelief*, temporarily leading a user to accept the agent as an alive or a real character
  - e.g., characters in game and films

Mobility

- agent's *ability to change its location in the environment*, being it the *physical* world (robots), or the *virtual* worl
  - e.g., virtual agents, Internet network

Outline

- What is an agent?
- Intelligent agents and agent properties
- **Environment properties**
- Agent applications

Environment

Agent systems further characterized by *properties of the surrounding environment*

Sensors: define the *perceptors* for the agent perceive the world

Effectors: define the *actuators* for the agent perform in the world

Interaction with environment

- *sensors*: cameras, speed sensor, gps, sonar
- *actuators*: start engine, accelerate, break, turn, start light, change gears

Environment properties

Environment properties

- accessibility
- determinism
- dynamism
- continuity
- memory

Agents…

- typically have *only partial access and control over the environment*
- have to *decide what action to execute* in order to attain goals

Environment properties

Accessible vs. Inaccessible

- accessible environment: *agent can obtain complete, accurate, up-to-date data about the environment's state*
- most moderately complex environments are inaccessible

Environment properties

Deterministic vs. non-deterministic

- deterministic environment: *action has a single guaranteed effect*
- physical world is for us (humans) non-deterministic

Environment properties

Static vs. dynamic

- static environment: *world does not change while the agent is deliberating*

Discrete vs. continuous

- discrete environment: *fixed, finite number of possible actions and percepts*

Environment properties

Episodic vs. non-episodic

- Episodic environment: *world can be divided in a series of intervals (episodes) independent of each other*
- What happens in one episode has no influence on others

Outline

- What is an agent?
- Intelligent agents and agent properties
- Environment properties
- **Agent applications**

Agent applications

- Agent systems can exist in
  - *physical space*: robots
  - *cyberspace*: software and interface agents
  - *simulated physical space*: traffic simulators
  - *hybrid*: virtual agents interacting with humans

Agents in Games

AlphaGO

https://www.youtube.com/watch?v=8tq1C8spV_g

LLM/LWM ChatBots: ChatGPT, Gemini, DeepSeek

Thank You

rui.prada@tecnico.ulisboa.pt

TÉCNICO LISBOA

Introduction to multiagent systems

Perspectives on agent systems

Outline

- **Motivation**
- Agent and Multiagent Systems, a definition
- Examples of Agent-based Systems
- Challenges in Multiagent Systems
- Relevance of Multiagent Systems
- Frequently Asked Questions
- History and prospect

Motivation

- Five ongoing trends have marked the history of computing:
  - *ubiquity*;
  - *interconnection*;
  - *intelligence*;
  - *delegation*; and
  - *human-orientation*

Motivation

- ***Ubiquity:*** **We now see processing power in places and devices that would have once been uneconomic**

Motivation

- ***Interconnection :*** **Computer systems today no longer stand alone, but are networked into large distributed systems**

Motivation

- ***Intelligence:*** **We are now engineering complex systems that perform tasks that were unthinkable only a short time ago**

Motivation

- ***Delegation:*** **We are giving more and more control to computers (devices, robots, cars,…)**

Motivation

- ***Human-orientation:*** ***In the past, users had to program computers with low-level code. We now use computers with more human-oriented abstractions***

Where does it bring us?

- *Delegation* and *Intelligence* imply the need to build computer systems that can *act effectively on our behalf*

- This implies:
  - The ability of computer systems to act *independently*
  - The ability of computer systems to act in a way that *represents our best interests* while interacting with other humans or systems

Where does it bring us?

- *Interconnection* and *Distribution* have become a very important topic in Computer Science
- But Interconnection and Distribution, coupled with the *need for systems to represent our best interests*…
- Implies systems that can *cooperate* and *reach agreements* (or even *compete*)

Where does it bring us?

- All these trends have led to the emergence of a new field in Computer Science: *multiagent systems*

Outline

- Motivation
- **Agent and Multiagent Systems, a definition**
- Examples of Agent-based systems
- Challenges in Multiagent Systems
- Relevance of Multiagent Systems
- Frequently Asked Questions
- History and prospect

Agents, a Definition

- An agent is a computer system that is capable of *independent (autonomous)* action on behalf of its user or owner

- *Autonomy* is a key aspect of an agent!
  - An agent should figure out what to do
  - Rather than constantly being told what to do

Multiagent Systems, a Definition

- A multiagent system is one that consists of a *set of agents*, which *interact* with one another
- In the most general case, agents will be *acting on behalf of users with different goals and motivations*
- To successfully interact, they will require the ability to *cooperate*, *coordinate*, and *negotiate* with each other, much as people do

The two key problems

- The course covers two key problems:
  - *Agent design: How do we build agents?*
    - Agents that are independent, autonomous, and able to carry out tasks we delegate to them
    - Decision making is a key aspect!
  - *Society design: How do we build agents that are capable of interacting?*
    - cooperating, coordinating, negotiating with other agents
- These are the ***micro*** and ***macro*** perspectives

Outline

- Motivation
- Agent and Multiagent Systems, a definition
- **Examples of Agent-based systems**
- Challenges in Multiagent Systems
- Relevance of Multiagent Systems
- Frequently Asked Questions
- History and prospect

Examples of Agents

- *Autonomy - NASA's Mars 2021 Perseverance rover*

https://www.youtube.com/watch?v=M4tdMR5HLtg

Examples of Agents

- *Cooperation and Coordination - RoboCup*

https://www.youtube.com/watch?v=_Y5_iGxWFrQ

Examples of Agents

- *Trading Agent – Automated/Algorithmic Trading*

https://www.youtube.com/watch?v=OPm_EDTrz7Y

Examples of Agents

- *Unitree Dancing Robots*

https://www.youtube.com/watch?v=Fw_dSNxhhY4

Examples of Agents

- *Business Process Automation – Robotic Process Automation*

https://www.youtube.com/watch?v=9URSbTOE4YI

Forbes

INNOVATION > ENTERPRISE TECH

Agentic AI: The Next Big Breakthrough That's Transforming Business And Technology

By Bernard Marr, Contributor.

Follow Author

Sep 06, 2024, 01:50am EDT

Share Save Comment 1

Agentic AI is poised to revolutionize how we interact with artificial intelligence, promising more ... More
ADOBE STOCK

https://www.forbes.com/sites/bernardmarr/2024/09/06/agentic-ai-the-next-big-breakthrough-thats-transforming-business-and-technology/

Outline

- Motivation
- Agent and Multiagent Systems, a definition
- Examples of Agent-based Systems
- **Challenges in Multiagent Systems**
- Relevance of Multiagent Systems
- Frequently Asked Questions
- History and prospect

Challenges in Multiagent Systems

- In Multiagent Systems, we address questions such as:
  - How can agents *interact with each other*?
  - How can *cooperation emerge in societies* of self-interested agents?
  - What kinds of *languages can agents use to communicate*?

Challenges in Multiagent Systems

- In Multiagent Systems, we address questions such as:
  - How can *self-interested agents recognize conflict*, and how can they (nevertheless) *reach agreement*?
  - How can autonomous agents *coordinate their activities* to *cooperatively achieve goals*?

Multiagent Systems

- While *these questions are all addressed in part by other disciplines* (notably economics and social sciences)

- What makes the multiagent systems field unique is that it emphasizes that the *agents in question are computational, information processing* entities.

Outline

- Motivation
- Agent and Multiagent Systems, a definition
- Examples of Agent-based Systems
- Challenges in Multiagent Systems
- **Relevance of Multiagent Systems**
- Frequently Asked Questions
- History and prospect

On the relevance of agent systems

Many different views of what multiagent systems are:

- Agents as a paradigm for *software engineering*
- Agents as a tool to *understand societies*
- Agents as a way to search for *theoretical foundations*
- Role of *agents in other sciences*

Agents as a paradigm for software engineering

Engineering multiagent systems = Engineering complex software

- Many dynamically interacting components
- Decentralized approach
- Unforeseen situations
- Fault-tolerant (one component fails, others still alive)
- Adaptive/Flexible behavior

"*interaction* is probably the most important single characteristic of complex software"

Agents as a tool to understand societies

New tool for simulating society

- multiagent based simulations
- understand dynamics/behavior

Agents as a way to search for theoretical foundations

- Derive formal properties from single/multi agent behavior
- Theorem proofs

Multi-agents is interdisciplinary

- Economics/game theory
  - Interactions among self-interested agents / economic entities
  - Rational agents able to mimic humans / organizations

- Social sciences
  - Interested in MAS to model/simulated social behavior
  - Model of emotions and their impact
  - Inspiration from agent traits

Multi-agents is interdisciplinary

Influenced by and influences many other fields:

- *ecology*
- *ethology*
- *philosophy*
- *logic*
- *psychology*
- *sociology*
- *cognitive science*
- *anthropology*

Strength: one can use well-founded methodologies
Weakness: many different views as to what an agent is about

Outline

- Motivation
- Agent and Multiagent Systems, a definition
- Examples of Agent-based Systems
- Challenges in Multiagent Systems
- Relevance of Multiagent Systems
- **Frequently Asked Questions**
- History and prospect

Agents and Artificial Intelligence

Isn't it all just artificial intelligence?

Isn't building an agent what AI is all about?

- AI is largely concerned with the *components of intelligence*
  - Ability to learn, plan, act, etc.

- Classical AI ignores the *social* aspects of agency
  - Ability to communicate, coordinate, cooperate, and reach agreements

Agents and Distributed Systems

Isn't it all just Distributed/ Concurrent Systems?

- There is much to learn from this community, but:
  - Agents are assumed to be autonomous
  - Agents are (can be) self-interested

Agents and Economics

Isn't it all just Economics/Game Theory?

- These fields also have a lot to teach us in multiagent systems, but:
  - Many concepts in Game Theory (e.g., Nash equilibrium) were developed without a view to computation
  - Some assumptions in economics/game theory (such as a rational agent) may not be valid or useful in building artificial agent societies

Outline

- Motivation
- Agent and Multiagent Systems, a definition
- Examples of Agent-based Systems
- Challenges in Multiagent Systems
- Relevance of Multiagent Systems
- Frequently Asked Questions
- **History and prospect**

History

- First conference *Workshop on Distributed Artificial Intelligence* in **1980**
- MAAMAW in Europe in **1980** (after launch in European Conference on AI)
- First international meeting ICMAS in **1995**
- *Workshop Agent Theories, Arch. and Languages* (ATAL) launched in ECAI in **1994**
- Finally: *Autonomous Agents Conference* held in 1997-99 (US) and 2000 in Europe
- In **2002**: ICMAS and AA merged to launch the largest conf. on agents: **AAMAS**

Thank You

rui.prada@tecnico.ulisboa.pt

Agent Architectures

Outline

- **Introduction to agent architectures**
- Abstract architectures for agents
- Deductive reasoning agents
- Agents as intentional systems
- Reactive agents
- Hybrid architectures

Recalling concepts

AGENT & ENVIRONMENT

- **Sense – Decide - Act**
- **Agent properties** (autonomous, reactive, pro-active, social, adaptive, etc.)
- **Social Ability** (cooperation, coordination, negotiation)
- **Environment properties** (accessibility, determinism, continuity, etc)

The architectural stance

Modular thinking in Computer Science

- <u>System components</u>
  - perception
  - decision making
    - planning
    - learning
  - action

⇒ *an architecture, please!*

- <u>Component interactions</u>
  - between internal components
  - between agent and environment
  - between agents

Agent architectures

- **What is an agent architecture?**
  - Principles describing agent behavior with
    - formal/abstract view of agents
    - map of the internals (control-flow)

- **What is the goal of an architecture?**
  - Guide agent design and engineering

Agent architectures

- **What is an abstract architecture?**
  - Common principles describing agent behavior independently of their specificities
  - A template for the construction of agents

Outline

- Introduction to agent architectures
- **Abstract architectures for agents**
- Deductive reasoning agents
- Agents as intentional systems
- Reactive agents
- Hybrid architectures

Abstract architectures for agents

Let us make formal the abstract view of agents:

- ***Environment states***: (*finite*) set of (*discrete*) *states*

  $E = \{e_0, e_1, \dots\}$

- *Agents have a set of* ***actions*** *(which transforms the environment's state) :*

  $Ac = \{\alpha_0, \alpha_1, \dots\}$

- ***Run****:* finite sequence of interleaved *states* and *actions*

  $r: e_0 \xrightarrow{\alpha_0} e_1 \xrightarrow{\alpha_1} e_2 \xrightarrow{\alpha_2} \dots \xrightarrow{\alpha_{n-1}} e_n$

Abstract architectures for agents

Let:

- R be the set of **all such possible runs** (over E and Ac)
  where r, r' are members of R

- R^{Ac} is the subset of these that *end with an action*

- R^{E} is the subset of these that *end with an environment state*

Abstract architectures for agents

- A **state transformer** (environment changes):

$$\tau: R^{Ac} \rightarrow \wp(E)$$

Maps a run (ending in an action) to a set of possible environment states

Important points about this definition:

- History dependent
- Non-determinism

if $\tau(r)=\emptyset$, there are no possible successor states to r (system has *ended* its run)

Abstract architectures for agents

- An **environment** can now be fully defined as a triple

  $Env = < E, e_0, \tau >$ where:

  – E is a set of environment states

  – $e_0 \in E$ is the initial state

  – τ is a state transformer function

Abstract architectures for agents

- **Agent** is a function which maps runs to actions:

$$Ag: R^E \rightarrow Ac$$

An **agent makes a decision** (i.e., action to perform) based on the **history** of system that it has witnessed to date (i.e., R^E)

- Let $\boldsymbol{AG}$ be the **set of all agents**, $Ag \in AG$

Abstract architectures for agents

- A **system** is a **pair with:**
  - **one (or more) agent(s)** and
  - **an environment**

- Any **system** has **a set of possible runs**

Abstract architectures for agents

- We denote the **set of runs of agent $\boldsymbol{Ag}$** in **Environment $\boldsymbol{Env}$** by

$$R(Ag, Env)$$

- We assume $R(Ag, Env)$ contains only **runs that have ended**

Outline

- Introduction to agent architectures
- Abstract architectures for agents
- **Deductive reasoning agents**
- Agents as intentional systems
- Reactive agents
- Hybrid architectures

The oldest agent architecture

- **1956–1985**: pretty much all agents designed within AI were ***symbolic reasoning*** agents (mostly *deductive reasoning*)
  - agents with *explicit logical reasoning* to decide what to do

- **1985–present**: problems with symbolic reasoning led to the so-called ***reactive*** *agents* movement (1985–present)

- **1990-present**: diverse alternatives, including ***hybrid*** architectures, combining the best of *deliberative* reasoning and *reactiveness*

Deductive reasoning agents

- Classical approach for creating deductive agents:
  - **Agent as a knowledge-based system**
- This paradigm is known as **symbolic AI**

Deductive reasoning agents

Two key problems to be solved:

1. ***Transduction problem***:
   - translating real-world environment into an accurate, adequate symbolic description.

Fields: computer vision, speech understanding, learning…

Deductive reasoning agents

Two key problems to be solved:

2. ***Representation/reasoning problem***:
   - symbolically representing information
   - how to get agents to reason with this information

Fields: knowledge representation, automated reasoning, planning

Deductive reasoning agents

- ***Deductive reasoning agent*** (architecture) is one that:
  - contains an explicit **symbolic representation of the world**
  - **internal state** given by formulae (**predicate logic**)

- Example with first-order predicate logic:

```
isopen(valve221)
temperature(reactor4726, 321)
pressure(tank776, 28)
```

Deductive reasoning agents

- ***Deductive reasoning agent*** (architecture):
  - **analogous to beliefs in humans**:
    - internal state may include incorrect/outdated info
  - **makes decisions via *symbolic reasoning***
    - proving theorems without breaking axioms on what is possible

Agents with state

Agent decision:

- D (internal state = set of formulae or database)

– $see: S \rightarrow Per$ (observe the environment)

– $next: D \times Per \rightarrow D$ (update internal state)

– $action: D \rightarrow Ac$ (decision making with deduction rules)

Deductive reasoning agents

How can an **agent decide what to do using theorem proving**?

- Use logic to encode a theory stating the *best* action to perform in a given situation

Deductive reasoning agents

- Let:
  - ρ be this theory (typically **deduction rules**)
  - *DB* be the logical data (database) describing **current state** of the world
  - Ac be the **set of actions** the agent can perform
  - $DB \vdash_{\rho} \phi$ means that **formula ϕ can be proved from database *DB* using deduction rules ρ**

Deductive reasoning agents

- Agent's action selection function (i.e., $action: D \rightarrow Ac$) is defined in terms of its deduction rules

```
function action(DB:D) returns an action Ac
begin
   /* for each action, attempts to prove Do(a) from its database using deduction rules */
   for each a ∈ Ac do
            if DB ⊢_ρ Do(a) then return a
   end for
   /* attempts to find an action such that ¬Do(a) cannot be derived (i.e., not explicitly forbidden) */
   for each a ∈ Ac do
            if DB ⊢_ρ ¬Do(a) = false then return a
   end for
   return null /* no action found */
end function
```

Vacuum world

Vacuum world

(0,2)	(1,2)	(2,2)
(0,1)	(1,1)	(2,1)
(0,0)	(1,0)	(2,0)

Agent starts here (facing north)

Vacuum world

- **Environment state**
  - S ={(0,0,$d_{0,0}$), (0,1,$d_{0,1}$), (0,2,$d_{0,2}$), (1,0,$d_{1,0}$), (1,1,$d_{1,1}$), (1,2,$d_{1,2}$), (2,0,$d_{2,0}$), (2,1,$d_{2,1}$), (2,2,$d_{2,2}$)}

- Agent can receive a **percept *dirt* or *null***
  - i.e., either **dirt under the agent** or not
  - Per= {*dirt*, *null*}

- **Actions**: *forward*, *suck*, or *turn* (right 90°)
  - Ac ={*forward*, *suck*, *turn*}

Vacuum world

- Internal state DB: three domain predicates
  - $In(x, y)$ – agent is at (x, y)
  - $Dirt(x, y)$ – there is dirt under the agent at (x, y)
  - $Facing(d)$ – the agent is facing direction d

Vacuum world

We need deduction rules for agent's behavior!

Vacuum world

- Deduction rules (agent's behavior):
  - $In(x, y) \wedge Dirt(x, y) \longrightarrow Do(suck)$
  - $In(0,0) \wedge Facing(north) \wedge \neg Dirt(0,0) \longrightarrow Do(forward)$
  - $In(0,0) \wedge \neg Facing(north) \wedge \neg Dirt(0,0) \longrightarrow Do(turn)$
  - $In(0,1) \wedge Facing(north) \wedge \neg Dirt(0,1) \longrightarrow Do(forward)$
  - $In(0,1) \wedge \neg Facing(north) \wedge \neg Dirt(0,1) \longrightarrow Do(turn)$
  - …

Final remarks: deductive reasoning agents

- **Agent's decision making strategy** – encoded as logical theory
- **Agent's action** – reduces to a problem of proof
- Logic-based approach are **elegant** and have **(clean) logical semantics**

Final remarks: deductive reasoning agents

- **Disadvantages:**
  - Inherent **computational complexity** of theorem proving
  - Cannot **operate effectively in time-constrained environments**
  - The **environment cannot change** while the agent is making a decision
  - Not easy to **represent and reason about complex and dynamic environments**

Outline

- Introduction to agent architectures
- Abstract architectures for agents
- Deductive reasoning agents
- **Agents as intentional systems**
- Reactive agents
- Hybrid architectures

Agents as Intentional Systems

Intentional stance

Develop **agent behaviors** in terms of ***mental states*** (beliefs, desires, wishes, hopes, …)

Agents as Intentional Systems

Examples:

- "Michael took his umbrella because he *believed* it was going to rain."

- "John worked hard because he *wanted* to obtain a PhD."

…such attitudes are called ***intentional notions***

Agents as Intentional Systems

This approach can be useful to model:

- **complex systems**
- systems whose **structure is incompletely known**

yes

Is it <u>legitimate</u> or <u>useful</u> to attribute beliefs, desires, and so on, to artificial agents?

How to implement agents using the Intentional Stance?

Intentional Systems are the base for deliberative agents,
which follow the *intentional stance* through practical reasoning

How to implement agents using the Intentional Stance?

- **Intentional systems** are the base for **deliberative agents**
  - This **agent architecture** has its origins in the **philosophical work** of Bratman:
    - Michael E. Bratman. *Intention, Plans and Practical Reason*. Harvard University Press, 1987.

How to implement agents using the Intentional Stance?

"Practical reasoning is a matter of ***weighing conflicting considerations*** **for and against competing options**, where the relevant considerations are provided by what the **agent** ***desires/values/cares*** **about and what the agent** ***believes***." [Bratman, 1990]

Practical reasoning

Practical Reasoning = Deliberation + Means-Ends Reasoning

- ***Deliberation****:* deciding *what* state of affairs an agent wants to achieve from (possibly conflicting) desires

- ***Means-Ends Reasoning****:* deciding *how* an agent wants to achieve these states of affairs

The B.D.I. model

- Bratman's philosophical work was used as an **inspiration for creating an architecture for intelligent agents** based on the mental attitudes of:
  - ***beliefs***
  - ***desires***
  - ***intentions***

The B.D.I. model

- **Beliefs**
  - Information about the environment, other agents, and itself
- **Desires**
  - Desires/goals are state of affairs to achieve
- **Intentions**
  - Commitments to achieving particular goals

Intentions in practical reasoning

Intentions are stronger than mere desires:

"My desire to play basketball this afternoon is merely a potential influencer of my conduct.

It must be viewed with my other relevant desires…

Once I intend to play basketball this afternoon, the matter is settled: I normally need not continue to weigh the pros and cons.

When the afternoon arrives, I will normally just proceed with my intentions." (Bratman, 1990)

Intentions in practical reasoning

And **Intentions drive means-ends reasoning**

- They *lead to action* because I *attempt to achieve* them
- I try to decide *how* to achieve them
- If one course of action *fails*, I usually *attempt others*

Intentions in practical reasoning

For example:

- *You might consider a career as an academic or a career in industry (deliberation)*
- *You have to decide the career (deliberation)*
- *Your decision is to be an academic (intention/state of affairs)*
- *You make a plan: apply for PhD program, get a PhD, etc.*
  - *You decide how to achieve the state of affairs (means-end reasoning)*

Intentions in practical reasoning

Property: **Intentions persist**

- *I do not give up without good reason*

E.g., If your intention is to become an academic, then you should persist with this intention

Intentions in practical reasoning

Property: **Intentions constrain future deliberation**

- *I will not entertain options that are inconsistent*

filter of admissibility

E.g., If I have an intention to write a book, so I cannot consider the option of partying every night

Intentions in practical reasoning

Property: **Intentions influence beliefs**

- *Intentions are closely related to beliefs about the future*

E.g., If you intend to become an academic, then you should believe that, assuming some background conditions, you will indeed become an academic

Deliberation and belief revision

So how do we model ***deliberation in agents***?

- **Belief revision function**
  - Update beliefs with sensory input and previous belief
- **Function to generate options**
  - Use beliefs and existing intentions to generate options (=desires)
- **Filtering function**
  - Choose between competing alternatives and commit to their achievement

Deliberation and belief revision

So how do we model ***deliberation in agents***?

- revise the agent's beliefs (belief revision function):

$$brf\colon 2^{Bel} \times Per \rightarrow 2^{Bel}$$

- produce the agent's desires/options (option generation function):

$$options\colon 2^{Bel} \times 2^{Int} \rightarrow 2^{Des}$$

- select the *best* option(s) for the agent to commit to (filter function):

$$filter\colon 2^{Bel} \times 2^{Des} \times 2^{Int} \rightarrow 2^{Int}$$

Means-Ends reasoning

An agent's means-ends function

$$plan\colon\ 2^{Bel} \times 2^{Int} \times 2^{Ac} \rightarrow Plan$$

Implementing a practical reasoning agent

Decision-making is a *loop*:

1. *Observe* the world and *update beliefs*
2. *Deliberate* to decide the *intention(s)*
   - determine available *options*
   - *filter*
3. Use *means-ends reasoning* to find a *plan* for the intention(s)
4. *Execute* the plan
5. Return to 1

Commitments

How committed an agent should be to its intention?

How long should an intention persist?

- A commitment implies ***temporal persistence***.
- But to what extent?

When do I give up pursuing an Intention?

Commitment strategies

- Blind commitment
- Single-minded commitment
- Open-minded commitment

Blind commitment

Maintains an intention *until* it believes the intention has been *achieved*

Aka *fanatical* commitment

Single-minded commitment

Maintains an intention *until* it believes that either:

- the intention has been *achieved*
- is *no longer possible* to achieve

Open-minded commitment

Maintains an intention *as long as*

- it has not been *achieved*
- it is still *desired*

Outline

- Introduction to agent architectures
- Abstract architectures for agents
- Deductive reasoning agents
- Agents as intentional systems
- **Reactive agents**
- Hybrid architectures

Reactive agents

- Many **problems** with **symbolic/logical approaches**, for example:
  - Inherent **computational complexity** of theorem proving
  - Cannot **operate effectively in time-constrained environments**

Reactive agents

- Many **problems** with **symbolic/logical approaches**, for example:
  - The **environment cannot change** while the **agent is making a decision**
  - **Not easy to represent and reason** about complex and dynamic environments

Reactive agents

- In the mid to late 1980s, researchers started to **investigate alternatives to symbolic AI paradigm**

- These **new approaches** had a few themes in common:
  - **Rejection of symbolic representation** and syntactic manipulation (e.g., logic programming)

Reactive agents

- These **new approaches** had a few themes in common:
  - The idea that **intelligent behavior** is **linked to the environment**
  - **Intelligent behavior can emerge** from the interaction of various simpler behaviors

Reactive agents

What are reactive agents?

- **Agents equipped with simple processing units** that perceive and quickly **react to changes** in the environment
- **Do not use complex symbolic reasoning**

Reactive agents

What are reactive agents?

- In reactive agent systems, **intelligence** is **not a property of a single component**
- The **intelligence is distributed** in the system and **emerges from the interaction** among agent components and the environment

Purely reactive agents

- **Purely reactive agents** make **no reference to their history**
  - no internal state
  - decision making entirely on the present

  - Formally:

$$Ag: E \rightarrow Ac$$

```
function decide(perception)
  current_state <- INTERPRET-INPUT(perception)
  rule <- RULE-MATCH(current_state,rules)
  action <- RULE-ACTION(rule)
return action
```

(Russell and Norvig, 1994)

Reactive architectures

Inspiration: **intelligent behavior** of animals in the world

- **simple behaviors** of each **individual agent**
- **complex behaviors** comes from **combining individual behaviors**

Brooks: subsumption architecture

Dr. Rodney Brooks' short bio:

- 1981: **PhD Stanford**
- 1984 - 2010: Professor at **MIT**
- 1997 – 2007: Director of **MIT Artificial Intelligence Lab**
- 1990 - 2011: Founder, Board Member, and CTO of **iRobot Corp**
- 2008 – 2018: Founder, Board Member, and CTO of **Rethink Robotics**
- 2019 - Present: Founder and CTO **of Robust.AI**

Dr. Brooks was one of the most vocal and influential critics of the symbolic approach

Brooks: subsumption architecture

Decision-making:

- A set of *task-accomplishing behaviors*
- Each **behavior module** can be seen as an **action selection function**

Brooks: subsumption architecture

- **Behavior module:**
  - **Perceptual inputs** are mapped into **actions**
  - Each **behavior module** is intended to **perform a task**

Brooks: subsumption architecture

- Behavior modules are finite-state machines

Subsumption architecture: layered control

- However, many behaviors can *fire simultaneously*

- Subsumption hierarchy:
  - behaviors arranged into layers
  - lower layers inhibit higher layers

Agent development using Brooks' architecture

1. **Create a module for a particular task**
   - should **link perception and action**
   - should work by itself

2. **Add more modules**
   - the **priorities** for the behaviors need to be **re-adjusted** every time **one module is added**

Agent development using Brooks' architecture

Common requirements:

1. Deal with **multiple goals**

2. Deal with **multiple sensors**
   - may provide conflicting data

3. Be **robust** in dealing with changes in the environment

4. Deal with **time constraints**

Advantages of reactive agents

- Simplicity
- Economy
- Computational tractability

Advantages of reactive agents

- Robustness against failure
- Elegance
- Extensibility

Outline

- Introduction to agent architectures
- Abstract architectures for agents
- Deductive reasoning agents
- Agents as intentional systems
- Reactive agents
- **Hybrid architectures**

Limitations of reactive agents

- Decisions are only based on local information: agents have a ***short-term* view**
- Agents ***need sufficient local information*** to make decisions
- **No learning**: how to guarantee reactive rules evolve?
- Not trivial to ***engineer*** agents with complex behavior
- **Not easy to predict complex behavior** when agents have a *high number of layers*

Criticisms to deliberative agents

- **Deliberation** and **Planning** with **incomplete info** can be a problem
- **Speed of decisions can be slow** to deal with the real world
- Many architectures rely on a **symbolic approach**
- The need to be **grounded to the real world**

Hybrid agents

Many researchers argue that **neither a completely deliberative nor completely reactive approach is suitable**

⇒ *hybrid* systems to marry classical and alternative approaches

Hybrid architectures

- Requirement: **agent must have both reactive and deliberative behaviors**
- Often, the **reactive subsystem is given some kind of precedence over the deliberative subsystem**
- This kind of structuring leads naturally to the **idea of a *layered* architecture**

Hybrid architectures

A key problem in hybrid architectures: **what kind of control flow should we consider between the agent's subsystems?**

- *Horizontal layering*
  Layers are directly connected to the sensory input and action output. Each layer itself acts like an agent, producing suggestions on what action to perform

- *Vertical layering*
  Sensory input and action output are each dealt with by at most one layer each

Hybrid architectures: horizontal layering

- **Advantages**: simple, distributed, fault-tolerant
- **Disadvantages**:
  - global behavior may not be coherent
  - difficult to avoid conflict between layers

Hybrid architectures: vertical layering

Vertical (1 pass control)

Vertical (2 passes control))

Comparing hybrid architectures

m possible actions suggested by each layer, *n* layers

(a) Horizontal layering

(b) Vertical layering (One pass control)

(c) Vertical layering (Two pass control)

complex decision making regarding the action output

not tolerant to layer failure

DARPA grand challenge[1]

Main goal of the challenge:

- develop an autonomous car (i.e., self-driving car):
  - capable of **traversing unrehearsed off-road terrain**
  - travel a **175 mile** long course through the **Mojave desert**
  - **take no more than 10 hours**

[1]https://en.wikipedia.org/wiki/DARPA_Grand_Challenge

DARPA grand challenge

- Teams
  - 2004 (1M$ prize): 107 teams (15 finalists, 0 finished)
  - 2005 (2M$ prize): 195 teams (23 finalists, 5 finished)

- **The route is kept secret** until 2h before the race
  - At this time they received a route description in RDDF format

Stanley – 2005 Winner

Volkswagen R5 (4WD)

- treats autonomous navigation as a software problem

Stanley: sensory equipment

Perception: roof rack that houses

- 5 SICK laser range finders
- camera for long range perception
- 2 RADAR sensors
- antennae: 1 for GPS, 1 for GPS compass, and 1 radio antennae

Stanley control

3 main actuators

- brakes
- throttle
- steering

Stanley architecture

Stanley in action

http://www.youtube.com/watch?v=M2AcMnfzpNg

Since then…

Advantages of hybrid architectures

- One of the **most used architectures** currently
- Allows for a **real-time response** combined with **goal oriented-behavior**
- Reactivity can be privileged in relation to deliberation
- **Knowledge** about the world can be subdivided into layers
  - different **levels of abstraction**

Disadvantages of hybrid architectures

- **Vertical layering**: bottlenecks and fault intolerance
- **Horizontal layering**: complexity of decision making
- **Interactions between layers** are difficult to program and to test
  - one will need to analyze all the possible interactions between these layers
  - **an integration problem**

Thank You

rui.prada@tecnico.ulisboa.pt

Multiagent decision making and Games in Normal Form

Outline

- **Multiagent decision making with Game theory**
- Normal-form games
  - Example – Prisoner's dilemma
- Strictly dominated action
- Solution Concepts
  - Iterated elimination of strictly dominated actions
  - Nash Equilibrium

Multiagent decision making

- We are now going to study **multiagent decision making**
  - Group of agents coexist within an environment
  - Agents make simultaneous decisions

Game Theory

- What is game theory?
  - Mathematical study of **interactions** among **independent, self-interested agents**
  - It has been **applied in many disciplines**, such as:
    - Economics, political science, biology, psychology, linguistics, and **computer science (multiagent decision making)**

Game Theory

- What is game theory?
  - Agents are considered **self-interested**
    - Each agent has a **description of states (outcomes) it likes**
    - The dominant theory for modeling agent's interests is **utility theory**

Game Theory

The success of an agent depends on the decisions of <u>all agents</u>

Game Theory

- Game theory was **originally designed for modelling economical interaction**

- Game theory is now an **independent field**
  - Solid mathematical foundation
  - Many applications

Game Theory

- Game theory is based on **two assumptions:**
  - Participating agents are **rational**
  - Agents reason **strategically**
    - They **take into account the other agent's decision** in their own decision making process

Outline

- Multiagent decision making with Game theory
- **Normal-form games**
  - Example – Prisoner's dilemma
- Strictly dominated action
- Solution Concepts
  - Iterated elimination of strictly dominated actions
  - Nash Equilibrium

Normal-Form Games

- Can be viewed as the **multiagent extension** of utility-based decision making
- Also known as **static game or strategic games** in some game theory books

Normal-Form Games

- **Definition (Normal-form game):** *A (finite, n-person) normal-form game is a* ***tuple*** $(\boldsymbol{N}, \boldsymbol{A}, \boldsymbol{u})$***,*** *where:*
  - $\boldsymbol{N}$ *is a* ***finite set of*** $\boldsymbol{n}$ ***players****, indexed by* i*;*
  - $A = A_1 \times \cdots \times A_n$*, where* $\boldsymbol{A_i}$ ***is a finite set of actions available to player/agent*** $\boldsymbol{i}$*. Each vector* $\boldsymbol{a} = (\boldsymbol{a_1}, \dots, \boldsymbol{a_n}) \in \boldsymbol{A}$ ***is called an action profile (or joint action)***
  - $u = (u_1, \dots, u_n)$*, where* $u_i : A \longmapsto \mathbb{R}$ *is a real-valued utility (or payoff) function for player* i*.*

Strategic game

- In summary:
  - **Each agent chooses a single action** and then **receives a payoff that depends on the joint action**
  - The joint action is the **outcome** of the game
  - Although payoffs are common knowledge, an **agent does not know the actions of the other agents**
    - The best an agent can do is to **predict** the actions of others
  - A game's **solution** is a **prediction of the outcome** using the assumptions that all agents are rational and strategic

Notation in Economics

- The **normal-form representation** of a game specifies:
  - *an* $\boldsymbol{n}$**-player game**
  - the **strategies** available to each player
    - Let $s_i \in S_i$ denote a strategy for player i, where S_i is the strategy space
    - Let $(s_1, \dots, s_n)$ denote a combination of strategies
  - the **payoff** received by each player for each combination of strategies that could be chosen by the players
    - Let $u_i(s_1, \dots, s_n)$ denote player i's payoff function
  - Hence, we denote a **game** by $G = \{S_1, \dots, S_n, u_1, \dots, u_n)$

Outline

- Multiagent decision making with Game theory
- Normal-form games
  - **Example – Prisoner's dilemma**
- Strictly dominated action
- Solution Concepts
  - Iterated elimination of strictly dominated actions
  - Nash Equilibrium

Prisoner’s dilemma

- The **prisoner’s dilemma** is one of the **oldest and most studied model** in game theory

- Information about the game:

  - **Two suspects** are arrested and charged for a crime

  - The **police lack sufficient evidence** to convict the suspects, unless at least one confesses

  - The police hold the suspects in **separate cells**

Prisoner’s dilemma

- The **policemen explain the consequences** that will follow from the following actions:
  - **If neither confesses** then **both will be convicted** of a minor offense and sentenced to **one month** in jail
  - **If both confess** then **both will be sentenced** to jail for **six months**
  - **if one confesses but the other does not**, then the **confessor will be released** immediately but the **other will be sentenced** to **nine months** in jail

Prisoner's dilemma

- Elements of this normal-form game:
  - $\boldsymbol{N} =$ **{Prisoner 1, Prisoner 2}** (i.e., there are **2 agents** within the environment)
  - Each agent i can select an **action** a_i (or **strategy**) from his own action set A_i
    - $A_1 = A_2 = \{not\ confess, confess\}$
    - And $a = (a_1, a_2)$ is the **joint action** (or action/strategy profile)
      - e.g., $a = (a_1 = confess, a_2 = not\ confess)$
  - $u = (u_1(a_1, a_2), u_2(a_1, a_2))$, where u_i is the payoff function for each agent (next slide)

Prisoner's dilemma

- Elements of this normal-form game :
  - In the special case of two agents, the normal-form game can be represented by a **payoff matrix**

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

  - E.g.:
    - $u_1(a_1 = confess, a_2 = not\ confess) = 0$
    - $u_2(a_1 = confess, a_2 = not\ confess) = -9$

Prisoner's dilemma

- In its most general form, the Prisoner's Dilemma is any normal-form game as follows:

		Agent 2	
		Cooperate	***Defect***
Agent 1	***Cooperate***	a, a	b, c
	Defect	c, b	d, d

In which: $\boldsymbol{c > a > d > b}$

Outline

- Multiagent decision making with Game theory
- Normal-form games
  - Example – Prisoner's dilemma
- **Strictly dominated action**
- Solution Concepts
  - Iterated elimination of strictly dominated actions
  - Nash Equilibrium

Strictly dominated action

- **In game theory, we assume** that a **rational agent will never choose a suboptimal action** (or play a strictly dominated action/strategy)

- **Suboptimal action** will always result in lower payoffs for the agent than some other action, no matter what the other agents do

Strictly dominated action

- We formalize this concept as follows:
  - **Definition**: *We will say that an action a_i of agent i is* **strictly dominated** *by another action a_i' of agent i if*

$$u_i(a_i', a_{-i}) > u_i(a_i, a_{-i})$$

for all actions a_{-i} of the other agents

Strictly dominated action

- Example in the prisoner's dilemma:
  - **for prisoner 1,** the $not\ confess$ is a **strictly dominated action** by the action $confess$

$$u_1(a_1' = confess, a_2 = not\ confess) > u_1(a_1 = not\ confess, a_2 = not\ confess)$$
$$u_1(a_1' = confess, a_2 = confess) > u_1(a_1 = not\ confess, a_2 = confess)$$

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

Strictly dominated action

- Example in the prisoner's dilemma:
  - similarly, **for prisoner 2,** the $not\ confess$ is a **strictly dominated action** by the action $confess$

$$u_2(a_2' = confess, a_1 = not\ confess) > u_2(a_2 = not\ confess, a_1 = not\ confess)$$
$$u_2(a_2' = confess, a_1 = confess) > u_2(a_2 = not\ confess, a_1 = confess)$$

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

Outline

- Multiagent decision making with Game theory
- Normal-form games
  - Example – Prisoner's dilemma
- Strictly dominated action
- **Solution Concepts**
  - Iterated elimination of strictly dominated actions
  - Nash Equilibrium

Solution Concepts

- Now that we have a defined games in normal form:
  - How can we reason about these games?
  - How can we predict the outcomes of a game?
  - How can we predict the actions of a game?

Solution Concepts

- The problem of **reasoning about games** and **identifying certain subsets of outcomes** is called

Solution Concept

Outline

- Multiagent decision making with Game theory
- Normal-form games
  - Example – Prisoner's dilemma
- Strictly dominated action
- Solution Concepts
  - **Iterated elimination of strictly dominated actions**
  - Nash Equilibrium

Iterated elimination of strictly dominated actions

- The **iterated elimination of strictly dominated actions** is a **first solution concept** for games
  - This solution concept is based on the **assumption** that a **rational agent will never choose a suboptimal action** (or play a strictly dominated action/strategy)

Iterated elimination of strictly dominated actions

- The **iterated elimination of strictly dominated actions** is a solution technique that:
  - **iteratively eliminates strictly dominated actions** from all agents, until no more actions are strictly dominated
- This technique is based on **two assumptions**:
  - A rational agent would never take a strictly dominated action
  - It is *common knowledge* that all agents are rational

Iterated elimination of strictly dominated actions

- Example (**Game 2**):
  - **Imagine 2 agents with the following payoff matrix**
  - **Are there strictly dominated actions?**

		Agent 2		
		Left	***Middle***	***Right***
Agent 1	***Up***	1, 0	1, 2	0, 1
	Down	0, 3	0, 1	2, 0

Iterated elimination of strictly dominated actions

- Example (**Game 2**):
  - Let us first check agent 1
  - For agent 1, **neither Up nor Down is strictly dominated**

		Agent 2		
		Left	***Middle***	***Right***
Agent 1	***Up***	1, 0	1, 2	0, 1
	Down	0, 3	0, 1	2, 0

Iterated elimination of strictly dominated actions

- Example (**Game 2**):
  - Let us now check agent 2
  - For agent 2, ***Right*** **is strictly dominated by** ***Middle***

Agent 2

Agent 1		***Left***	***Middle***	***Right***
	Up	1, 0	1, 2	0, 1
	Down	0, 3	0, 1	2, 0

Iterated elimination of strictly dominated actions

- Example (**Game 2**):
  - A **rational** agent 2 will **not choose *Right***
  - Thus, if agent 1 knows that agent 2 is rational, then agent 1 can **eliminate *Right* from the action set**

Agent 2

Agent 1		***Left***	***Middle***	~~***Right***~~
	Up	1, 0	1, 2	~~0, 1~~
	Down	0, 3	0, 1	~~2, 0~~

Iterated elimination of strictly dominated actions

- Example (**Game 2**):
  - Hence, if agent 1 knows that agent 2 is rational, then **agent 1 can play the game as if it were the following:**

		Agent 2	
		Left	***Middle***
Agent 1	***Up***	1, 0	1, 2
	Down	0, 3	0, 1

Are there strictly dominated actions?

Iterated elimination of strictly dominated actions

- Example (**Game 2**):
  - ***Down* is now strictly dominated by *Up*** for agent 1.

		Agent 2	
		Left	***Middle***
Agent 1	***Up***	1, 0	1, 2
	Down	0, 3	0, 1

Iterated elimination of strictly dominated actions

- Example (**Game 2**):
  - If **agent 1 is rational** (and agent 1 knows that agent 2 is rational, so that the game below applies) then **agent 1 will not choose *Down***
  - if agent 2 knows that agent 1 is rational, then **agent 2 can eliminate *Down* from agent 1's action set**
    - Note that we assume that agent 2 knows that agent 1 knows that agent 2 is rational (so that agent 2 knows that the game below applies)

Agent 2

Agent 1	***Left***	***Middle***
Up	1, 0	1, 2
~~***Down***~~	~~0, 3~~	~~0, 1~~

Iterated elimination of strictly dominated actions

- Example (**Game 2**):
  - Hence, considering the rationality assumption, **the agents can play the game as if it were the following:**

		Agent 2	
		Left	***Middle***
Agent 1	***Up***	1, 0	1, 2

Are there strictly dominated actions?

Iterated elimination of strictly dominated actions

- Example (**Game 2**):
  - ***Left*** **is strictly dominated by** ***Middle*** **for agent 2**
  - **Thus, the solution (outcome) of this game is (*Up, Middle*)**

		Agent 2	
		~~***Left***~~	***Middle***
Agent 1	***Up***	~~1, 0~~	1, 2

Iterated elimination of strictly dominated actions

Can we use this solution concept to find the outcome in the Prisoner’s Dilemma?

Iterated elimination of strictly dominated actions

- Example with the Prisoner's Dilemma:
  - **Are there strictly dominated actions? Yes!**
  - For prisoner 1, ***Not confess*** **is strictly dominated by** ***Confess***

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

Iterated elimination of strictly dominated actions

- Example with the Prisoner's Dilemma:
  - A **rational** prisoner 1 will **not choose *Not confess***
  - Thus, if prisoner 2 knows that prisoner 1 is rational, then prisoner 2 can **eliminate *Not confess* from the action set**

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	~~-1, -1~~	~~-9, 0~~
	Confess	0, -9	-6, -6

Iterated elimination of strictly dominated actions

- Example with the Prisoner's Dilemma:
  - Hence, if prisoner 2 knows that prisoner 1 is rational, then **prisoner 2 can play the game as if it were the following:**

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Confess***	0, -9	-6, -6

Are there strictly dominated actions?

Iterated elimination of strictly dominated actions

- Example with the Prisoner's Dilemma:
  - ***Not confess*** **is strictly dominated by** ***Confess*** **for prisoner 2**
  - **Thus, the solution (outcome) of this game is (*Confess*, *Confess*)**

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Confess***	~~0, -9~~	-6, -6

Iterated elimination of strictly dominated actions

- Does the order matter in this algorithm?
- Will we end up with different outcomes if we change the order?

Iterated elimination of strictly dominated actions

- Example with the Prisoner's Dilemma:
  - **Recall that we started with this:**
    - For prisoner 1, ***Not confess*** **is strictly dominated by** ***Confess***

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

Iterated elimination of strictly dominated actions

- Example with the Prisoner's Dilemma:
  - **However, we could have started with this:**
    - For prisoner 2, ***Not confess*** **is strictly dominated by** ***Confess***

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

Iterated elimination of strictly dominated actions

- Example with the Prisoner's Dilemma:
  - A **rational** prisoner 2 will **not choose *Not confess***
  - Thus, if prisoner 1 knows that prisoner 2 is rational, then prisoner 1 can **eliminate *Not confess* from the action set**

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

Iterated elimination of strictly dominated actions

- Example with the Prisoner's Dilemma:
  - Hence, if prisoner 1 knows that prisoner 2 is rational, then **prisoner 1 can play the game as if it were the following:**

		Prisoner 2
		Confess
Prisoner 1	***Not confess***	-9, 0
	Confess	-6, -6

Iterated elimination of strictly dominated actions

- Example with the Prisoner's Dilemma:
  - ***Not confess*** **is strictly dominated by** ***Confess*** **for prisoner 1**
  - **Thus, the solution (outcome) of this game is (*Confess*, *Confess*)**

		Prisoner 2
		Confess
Prisoner 1	***Not confess***	~~-9, -9~~
	Confess	-6, -6

Iterated elimination of strictly dominated actions

- **Appealing idea** that rational agents do not play strictly dominated actions
- **First drawback:**
  - Each step requires a further assumption about what the agents know about each other's rationality
    - We need to assume that it is ***common knowledge*** that the **agents are rational**

Iterated elimination of strictly dominated actions

- **Second drawback:**
  - the process often produces a **very imprecise prediction** about the play of the game
    - For instance, if no actions are eliminated then anything could happen

Iterated elimination of strictly dominated actions

- **Second drawback:**
  - Does the game (i.e., **Game 3**) below **have any strictly dominated actions?**

	L	***C***	***R***
T	0, 4	4, 0	5, 3
M	4, 0	0, 4	5, 3
B	3, 5	3, 5	6, 6

Iterated elimination of strictly dominated actions

- **Second drawback:**

	L	*C*	*R*
T	0, 4	4, 0	5, 3
M	4, 0	0, 4	5, 3
B	3, 5	3, 5	6, 6

	L	*C*	*R*
T	0, 4	4, 0	5, 3
M	4, 0	0, 4	5, 3
B	3, 5	3, 5	6, 6

	L	*C*	*R*
T	0, 4	4, 0	5, 3
M	4, 0	0, 4	5, 3
B	3, 5	3, 5	6, 6

For agent 1, there are no strictly dominated actions

Iterated elimination of strictly dominated actions

- **Second drawback:**

	L	*C*	*R*
T	0, 4	4, 0	5, 3
M	4, 0	0, 4	5, 3
B	3, 5	3, 5	6, 6

	L	*C*	*R*
T	0, 4	4, 0	5, 3
M	4, 0	0, 4	5, 3
B	3, 5	3, 5	6, 6

	L	*C*	*R*
T	0, 4	4, 0	5, 3
M	4, 0	0, 4	5, 3
B	3, 5	3, 5	6, 6

For agent 2, there are no strictly dominated actions

Iterated elimination of strictly dominated actions

- **Second drawback:**
  - Hence, **Game 3** (below) **does not have any strictly dominated actions**
    - And the technique **does not produce a prediction whatsoever about this game, thus we are unsure about the outcome**

	L	***C***	***R***
T	0, 4	4, 0	5, 3
M	4, 0	0, 4	5, 3
B	3, 5	3, 5	6, 6

Outline

- Multiagent decision making with Game theory
- Normal-form games
  - Example – Prisoner's dilemma
- Strictly dominated action
- Solution Concepts
  - Iterated elimination of strictly dominated actions
  - **Nash Equilibrium**

Nash equilibrium

- The **Nash equilibrium** is a **stronger solution concept** than the iterated elimination of strictly dominated actions

- Hence, it produces **more accurate predictions in a wider class of games**

Nash equilibrium

- **Defintion**: *A **Nash equilibrium** is a joint action a^* with the property that the following holds for every agent i:*

$$u_i(a_i^*, a_{-i}^*) \geq u_i(a_i, a_{-i}^*)$$

for all action $a_i \in A_i$

- In other words, a Nash equilibrium is a **joint action from where no agent can *unilaterally* improve his payoff**
  - Hence, **no agent has any incentive to deviate**

Nash equilibrium

- Note that:
  - The **iterated elimination of strictly dominated actions** produces a solution of a game by means of an **algorithm**
  - However, the previous definition of **Nash equilibrium** describes a solution in terms of the **conditions that hold at that solution**

Nash equilibrium

- An alternative for the previous definition makes use of the **best-response function**

- **Definition:** *The **best-response function** for agent i is*

$$B_i(a_{-i}) = \{a_i \in A_i\colon\ u_i(a_i, a_{-i}) \geq u_i(a_i', a_{-i}) \text{ for all } a_i' \in A_i\}$$

- Note that $B_i(a_{-i})$ can be a set containing many actions

Nash equilibrium

- For instance, in the prisoner's dilemma:
  - If **prisoner 2** takes the action ***Not confess***
  - The **best response** of **prisoner 1** is the action ***Confess***
    - $B_1(a_2 = Not\ confess) = Confess$

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

Nash equilibrium

- We can compute the **best-response function for each agent**:
  - $B_1(a_2 = Not\ confess) = Confess$
  - $B_1(a_2 = Confess) = Confess$
  - $B_2(a_1 = Not\ confess) = Confess$
  - $B_2(a_1 = Confess) = Confess$

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, <u>0</u>
	Confess	<u>0</u>, -9	<u>-6</u>, <u>-6</u>

Nash equilibrium

- Using the definition of best-response function, we can now formulate an **equivalent definition** (to the first definition):

- **Definition 2**: *A **Nash equilibrium** is a joint action a^* with the property that the following holds for every agent i:*

$$a_i^* \in B_i(a_{-i}^*)$$

- In other words, in a Nash equilibrium, **each agent's actions is an optimal response to the other agents' actions**

Nash equilibrium

- We can compute the **best-response function for each agent**:
  - $B_1(a_2 = Not\ confess) = Confess$
  - $B_1(a_2 = Confess) = Confess$
  - $B_2(a_1 = Not\ confess) = Confess$
  - $B_2(a_1 = Confess) = Confess$

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

Nash Equilibrium

Nash equilibrium

- In the prisoner's dilemma, note that:

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

No incentives to deviate

Nash equilibrium

- So, we could use a **brute-force algorithm** to finding a game's **Nash equilibria** (NE):
  - Check whether each possible joint action satisfies the condition in the definition

Nash equilibrium

- Let us try this approach in **Game 2**:

		Agent 2		
		Left	***Middle***	***Right***
Agent 1	***Up***	1, 0	1, 2	0, 1
	Down	0, 3	0, 1	2, 0

Nash equilibrium

- In a two-agent game, this approach has the following steps:
  - for each agent and for each action of that agent, **determine the other agent's best response** to that action

		Agent 2		
		Left	***Middle***	***Right***
Agent 1	***Up***	1, 0	1, 2	0, 1
	Down	0, 3	0, 1	2, 0

Nash equilibrium

- In a two-agent game, this approach has the following steps:
  - A **pair of actions satisfies condition in the definition of a NE** if each agent's action is a best response to the other's
    - Thus, we have **NE if both payoffs are underlined in the corresponding cell of the payoff matrix**

Agent 2

Agent 1		***Left***	***Middle***	***Right***
	Up	1, 0	1, 2	0, 1
	Down	0, 3	0, 1	2, 0

Nash equilibrium

Nash equilibrium

- Let us try this approach in **Game 3**:

		Agent 2		
		L	***C***	***R***
	T	0, 4	4, 0	5, 3
Agent 1	***M***	4, 0	0, 4	5, 3
	B	3, 5	3, 5	6, 6

Nash equilibrium

- In a two-agent game, this approach has the following steps:
  - for each agent and for each action of that agent, **determine the other agent's best response** to that action

		Agent 2		
		L	***C***	***R***
	T	0, 4	4, 0	5, 3
Agent 1	***M***	4, 0	0, 4	5, 3
	B	3, 5	3, 5	6, 6

Nash equilibrium

- In a two-agent game, this approach has the following steps:
  - A **pair of actions satisfies condition in the definition of a NE** if each agent's action is a best response to the other's
    - Thus, we have **NE if both payoffs are underlined in the corresponding cell of the payoff matrix**

		Agent 2		
		L	***C***	***R***
	T	0, 4	4, 0	5, 3
Agent 1	***M***	4, 0	0, 4	5, 3
	B	3, 5	3, 5	6, 6

Nash equilibrium

Final remarks

- Although the two definitions suggest **a brute-force method** for finding the **Nash equilibrium** of a game
  - the **cost** of such an algorithm **is exponential in the number of agents**

Final remarks

- Addressing the **relation** between **Nash equilibrium (NE)** and **iterated elimination of strictly dominated actions (IESDA)**
  - Recall the **NE joint actions** in the Prisoners' Dilemma (*Confess, Confess*) and Game 2 (*Up*, *Middle*)
    - These are the only joint actions that ***"survived"*** **IESDA**
  - In fact, this **result can be generalized:**
    - If IESDA eliminates all but a single joint action a, then this joint action is the unique NE of the game

Final remarks

- Addressing the **relation** between **NE** and **IESDA:**
  - Since IESDA **frequently does not eliminate** all but a single joint action
    - We say that the **Nash equilibrium is a stronger solution concept** than iterated elimination of strictly dominated actions
  - For example in Game 3:
    - any outcome is possible with IESDA
    - while NE gives a unique prediction

Thank You

rui.prada@tecnico.ulisboa.pt

Multiagent decision making and Games in Normal Form - Exercises

Outline

- **Exercises IESDA**
- Exercises NE

Exercise 1

- Predict the outcome with the **iterated elimination of strictly dominated actions:**

		Agent 2		
		Left	***Right***	***Stay***
Agent 1	***Up***	1, 0	1, 2	0, 1
	Down	0, 3	0, 1	1, 2
	Stay	2, 4	2, 1	2, 3

Exercise 2

- **ADVERTISING** Scenario:
  - **Two companies** share a market, in which they currently make $5,000,000 each.
  - Both need to determine whether they should advertise.
  - For each company, advertising costs $2,000,000 and captures $3,000,000 from the competitor provided the competitor doesn't advertise.
  - What should the companies do?

Exercise 2

- How many agents?
- What are the action sets?
- What are the payoffs?
- Predict the outcome with the **iterated elimination of strictly dominated actions**

Outline

- Exercises IESDA
- **Exercises NE**

Exercise 1

- Predict the outcome using the **Nash Equilibrium definition:**

		Agent 2		
		Left	***Right***	***Stay***
Agent 1	***Up***	1, 0	3, 2	0, 1
	Down	0, 3	0, 1	1, 2
	Stay	-1, 4	2, 1	2, 3

Exercise 2

- When a single company controls all of the production of a single good, we call it a **monopoly**
- When exactly two competing firms control the production of an identical good, we call this economic environment a **duopoly**
  - A special case of an **oligopoly**
  - Each firm's production can affect the other's ability to profit

Exercise 2

- Suppose Firm 1 and Firm 2 must spend $1 to produce a unit of a good
- Consumer demand determines the price of the good
  - If the quantity available increases, then the price decreases
  - Let P be the consumer's market price of the good
  - Let Q be the total quantity of units produced by the two firms
    - Where $Q = Q_1 + Q_2$

Exercise 2

- If the firms collectively produce 6 or less units, then price demand function is the following: $P = 12 - 2Q = 12 - 2(Q_1 + Q_2)$

- If the firms collectively produce 6 or more, then $P = 0$

- Hint: each firm has only six plausible production choices: 0, 1, 2, 3, 4, 5 and the profit of firm i is equal to $(P - 1)Q_i$

Exercise 2

- How many agents?
- What are the action sets?
- What are the payoffs?
- Predict the outcome with the **definition of Nash Equilibrium**

Thank You

rui.prada@tecnico.ulisboa.pt

Rational agents

Outline

- **Rational agents and decision making**
- Utility theory for decision making
  - Binary relations
  - Preferences
  - Utility
- Making decisions
- Example

Rational Agents

- Let us recall the following agent property:

Rationality

- **Agent's ability** to **act** (i.e., make decision) in a way that **maximizes some utility function**

Rational Agents

- But what is a **utility** function?
- How can we use a **utility** function to make decisions?

Rational Agents

- Before we analyze agents…
- How do we (humans) make decisions?

Making decisions

- **How do we (humans) make decisions?**

- **Example 1:**
  - You won the lottery

  - You have the following decision to make. Either:
    - (Decision1) you receive your prize **today** (EUR 1,000,000.00)
    - (Decision2) you receive **next month**

- What is your decision?

Making decisions

- **Example 2:**
  - You have a plane ticket to Madeira (EUR 100)
  - The flight is overbooked

Making decisions

- The airline must ask for people to volunteer not to fly 'in exchange for benefits’.
- You have the following decision to make. Either, you choose:
  - (Decision1) a rerouting option + EUR 100 in cash
  - (Decision2) to keep your plane ticket (not volunteer)
- What is your decision?

Making decisions

- **Example 3:**
  - You are planning a trip for your next vacation
  - You have the following decisions:
    - (Decision1) Go to Hawaii (EUR 900)
    - (Decision2) Go to Cancun (EUR 500)
  - What is your decision?

Hawaii

Cancun

Making decisions

- While the decision in Example 1 is straightforward
  - I want my money now!

- Decision in Example 2 and 3 depend on preferences. Hence, this can lead to different outcomes.

  - **Many decisions are based on personal preferences!**

Making decisions

Key questions:

- How can I tell an agent what are my “preferences”?
- Can I treat decision making algorithmically?

Outline

- Rational agents and decision making
- Utility theory for decision making
  - **Binary relations**
  - Preferences
  - Utility
- Making decisions
- Example

Bibliography

UTILITY THEORY
FOR
DECISION MAKING

PETER C. FISHBURN
Research Analysis Corporation

JOHN WILEY & SONS, INC
NEW YORK · LONDON · SYDNEY · TORONTO

Binary relations

- A binary relation R on a set of outcomes Y is a set of ordered pairs (x, y) with

$$x, y \in Y$$

- We can also write this binary relation as follows:

$$xRy$$

Binary relations

- Examples of a binary relation
  - Let R_1 mean "is shorter than"
  - John (x) is 1.75m and Harry (y) is 1.85m
  - Then we can say that:

$$(xRy, \text{not } yRx)$$

Some binary relation properties

- Reflexive $\quad$ if xRx for every $x \in Y$
- Irreflexive $\quad$ if not xRx for every $x \in Y$
- Symmetric $\quad$ if $xRy \implies yRx,$ for every $x, y \in Y$
- Asymmetric $\quad$ if $xRy \implies$ not $yRx,$ for every $x, y \in Y$
- Antisymmetric $\quad$ if $(xRy, yRx) \implies x = y,$ for every $x, y \in Y$

Some binary relation properties

- Transitive $$\text{if } (xRy, yRz) \implies xRz, \text{ for every } x, y, z \in Y$$

- Negatively transitive

$$\text{if } (\text{not } xRy, \text{not } yRz) \implies \text{not } xRz, \text{ for every } x, y, z \in Y$$

- Connected or Complete $$\text{if } xRy \text{ or } yRx (\text{possibly both}) \text{ for every } x, y \in Y$$

- Weakly connected $$\text{if } x \neq y \implies (xRy \text{ or } yRx) \text{ throuhout } Y$$

Some binary relation properties

- Relation “is shorter than” is
  - Irreflexive $$\text{if not } xRx \text{ for every } x \in Y$$

    informally: a person cannot be shorter than himself

  - Asymmetric $$\text{if } xRy \implies \text{not } yRx, \text{ for every } x, y \in Y$$

    informally: if person 1 is shorter than person 2 then person 2 is not shorter than person 1

  - Transitive $$\text{if } (xRy, yRz) \implies xRz, \text{ for every } x, y, z \in Y$$

    informally: if person 1 is shorter than person 2 and person 2 is shorter than person 3 then person 1 is shorter than person 3

Outline

- Rational agents and decision making
- Utility theory for decision making
  - Binary relations
  - **Preferences**
  - Utility
- Making decisions
- Example

Preferences

- **Strict preference** is a **binary relation** on the set of outcomes, such that

$$x \succ y$$

denotes the proposition that

$\boldsymbol{x}$ **is preferred to** $\boldsymbol{y}$ (or x is better than y)

- We can also use the strict preference to express:

$$x \prec y$$

$\boldsymbol{y}$ **is preferred to** $\boldsymbol{x}$ (or y is better than x)

Preferences

- We can also define **indifference** as the absence of preference

$$x \sim y \iff (\text{not } x \prec y, \text{ not } x \succ y)$$

the two outcome are indifferent

(or x is neither better nor worse than y)

- Indifference might arise in the following situations:
  - One might feel that there is no difference between the outcomes
  - One is uncertain about his preferences

Preferences

- We can also define **preference-indifference** as the union of strict preference and indifference

$$x \preceq y \iff (x \prec y \text{ or } x \sim y)$$

x* is not better than *y

- Or

$$x \succeq y \iff (x \succ y \text{ or } x \sim y)$$

x* is not worse than *y

Properties of Preferences

- **Strict preference** $\succ$
  - antisymmetric, transitive, and negatively transitive
- **Indifference** $\sim$
  - reflexive, symmetric, and transitive
- **Preference-indifference** $\succeq$
  - complete and transitive

Rational preference

- A **rational preference** is a binary relation if:
  - complete and transitive

- The **preference-indifference** is complete and transitive
  - Hence a **rational preference**

Outline

- Rational agents and decision making
- Utility theory for decision making
  - Binary relations
  - Preferences
  - **Utility**
- Making decisions
- Example

Utility

- **Why don't we use (or code) preferences in our agents?**
  - From a computation perspective, they are cumbersome to maintain

- Recall that preferences express an ordering between outcomes
  - Thus, we can express the preferences with an **order-preserving function**

Utility

- **Does this order-preserving function exist?**
  - **Yes, if the preferences are rational**
  - When preferences are rational, **we can sort all outcomes consistently**

Utility

- **Theorem**:

Let X be a set of possible outcomes, and $\succeq$ a rational prefernce on X. Hence, there is a function $u : X \to \mathbb{R}$ such that $u(x) \geq u(y)$ if and only if $x \succeq y$, for all $x, y \in X$

We call u the utility function

Outline

- Rational agents and decision making
- Utility theory for decision making
  - Binary relations
  - Preferences
  - Utility
- **Making decisions**
- Example

Making Decisions

- Agents can use utility to make decisions:
  - Let A be a **set of actions**
  - Given $a \in A$, let $O(a)$ be an **outcome** when an agent selects action a
  - Hence, the **value of action** a is:

$$Q(a) \stackrel{\text{def}}{=} u(O(a))$$

Making Decisions

- So how can an agent make a decision?

$$\underset{a \in A}{\operatorname{argmax}}\, Q(a)$$

$$\underset{a \in A}{\operatorname{argmax}}\, u(O(a))$$

An agent selects an action with the maximum utility

UNCERTAINTY
AHEAD

Making Decisions Under Uncertainty

- So how can an agent make a decision?
  - Let O denote a finite **set of outcomes**
  - Given $o \in O$, let $P(o|a)$ denote the **probability of outcome** o when an agent selects action a
  - Hence, the **expected value of an action** is

$$Q(a) = \mathbb{E}[u(o)|a] = \sum_{o \in O} u(o)P(o|a)$$

Making Decisions Under Uncertainty

- So how can an agent make a decision?

$$\underset{a \in A}{\operatorname{argmax}}\, Q(a)$$

$$\underset{a \in A}{\operatorname{argmax}} \sum_{o \in O} u(o) P(o|a)$$

An agent selects an action with the maximum expected utility

Outline

- Motivation – making decisions
- Utility theory for decision making
  - Binary relations
  - Preferences
  - Utility
- Making decisions
- **Example**

Example: robot coffee machine

robot coffee machine

- O = {“coffee + mess”, “coffee + no mess”, “no coffee + no mess”}
  - set of outcomes

- A = {“get coffee”, “do nothing”}
  - set of actions

robot coffee machine

- P is the probability of an outcome

  - $\mathrm{P}(o = \text{“coffee + mess”} | a = \text{“get coffee”})$ = 0.2
  - $\mathrm{P}(o = \text{“coffee + no mess”} | a = \text{“get coffee”})$ = 0.8
  - $\mathrm{P}(o = \text{“no coffee + no mess”} | a = \text{“get coffee”})$ = 0

  - $\mathrm{P}(o = \text{“coffee + mess”} | a = \text{“do nothing”})$ = 0
  - $\mathrm{P}(o = \text{“coffee + no mess”} | a = \text{“do nothing”})$ = 0
  - $\mathrm{P}(o = \text{“no coffee + no mess”} | a = \text{“do nothing”})$ = 1.0

Decision Tree

robot coffee machine

- Now let us assume a different utility function:
  - $u(s = \text{“coffee + mess”})$ = 5
  - $u(s = \text{“coffee + no mess”})$ = 10
  - $u(s = \text{“no coffee + no mess”})$ = 0

I love coffee!

Decision Tree

Final remarks

- We have only considered **decision-making problems** that has **ONE agent**
- What if our environment has **two or more agents**?
  - **Two or more utility-maximizing agents** whose actions can **affect each other's utility**
  - We need a decision-making framework: **GAME THEORY!**

Thank You

rui.prada@tecnico.ulisboa.pt

Multiagent decision making and Games in Normal Form (Part 2)

Outline

- **Application of a Nash Equilibrium: Duopoly**
  - **Cournot Model**
- Mixed Strategy
- Exercise
- Final remarks - Mixed Strategy

Cournot Model – Application of a NE

- Some of the **earliest applications** of game theory
- The model focuses on the analyses of an **oligopoly**
- The model was created by **Antoine Augustin Cournot** in **1838**
  - **An application of a Nash equilibrium**
  - **A century before John Nash's equilibrium (1950)**

Cournot Model

- **Features** of the Cournot model:
  - There is **more than one firm** in the market
    - The number of firms is fixed
    - We will focus on the scenario with 2 firms (duopoly)
  - All firms produce a **homogeneous product**
    - In other words, there is no product differentiation
  - Firms **do not cooperate**
    - In other words, there is no collusion (or coalition)

Cournot Model

- **Features** of the Cournot model:
  - Firms have **market power**
    - In other words, each firm's output decision affects the product’s market price
  - Firms **choose quantities simultaneously**
    - One-shot game
    - Firms **compete in quantities**
  - The **firms are economically rational and act strategically**
    - Seeking to maximize profit given their competitors' decisions

Cournot Model

- Let q_1 and q_2 denote **the quantities** (of a homogeneous product) produced by firms 1 and 2, respectively

- Let $P(Q) = a - Q$ be the **market-clearing price** when the aggregate quantity on the market is $Q = q_1 + q_2$.

  - More precisely:
    - $P(Q) = a - Q$ for $Q < a$
    - $P(Q) = 0$, for $Q \geq a$

Cournot Model

- Assume that the **total cost to firm $\boldsymbol{i}$** of producing quantity q_i is $C_i(q_i) = cq_i$
  - there are **no fixed costs** and the **marginal cost is constant at $\boldsymbol{c}$**, where we assume $c < a$

- Assume that the **firms choose their quantities simultaneously**

Cournot Model

- Normal-form game:
  - How many agents?
  - What are the action sets?
  - What are the payoffs?

Cournot Model

- **How many agents?**
  - There are (of course) two agents in any duopoly game
    - i.e., two firms

Cournot Model

- **What are the action sets?**
  - the **actions available** to each firm are **the different quantities** it might produce
    - We will assume that **output** is **continuously divisible**
    - Naturally, **negative outputs are not feasible**.
    - Thus, each firm's **action set** can be represented as $A_i = [0, \infty)$ (the nonnegative real numbers)
      - A typical action a_i is a quantity choice $q_i \geq 0$

Cournot Model

- **What are the payoffs?**
  - The firm's payoff is its profit:

$$\pi_i(q_i, q_j) = q_i[P(q_i + q_j) - c] = q_i[a - (q_i + q_j) - c]$$

Cournot Model

- **How do we find the Nash equilibrium?**
  - The quantity pair (q_1^*, q_2^*) is a Nash equilibrium if, for each firm i, q_i^* solves

$$\max_{0 \le q_i \le \infty} \pi_i\left(q_i, q_j^*\right) = \max_{0 \le q_i \le \infty} q_i\left[a - (q_i + q_j^*) - c\right]$$

Cournot Model

- How do we find the Nash equilibrium?
  - Let us start with firm 1 and solve $\max_{0 \le q_1 \le \infty} q_1[a - (q_1 + q_2^*) - c]$:

$$\frac{\partial}{\partial q_1} q_1[a - (q_1 + q_2^*) - c] = 0$$

$$a - 2q_1^* - q_2^* - c = 0$$

$$q_1^* = \frac{1}{2}(a - q_2^* - c)$$

Cournot Model

- How do we find the Nash equilibrium?
  - And now for firm 2, we can solve $\max_{0 \le q_2 \le \infty} q_2[a - (q_2 + q_1^*) - c]$:

$$\frac{\partial}{\partial q_2} q_2[a - (q_2 + q_1^*) - c] = 0$$

$$a - 2q_2^* - q_1^* - c = 0$$

$$q_2^* = \frac{1}{2}(a - q_1^* - c)$$

Cournot Model

- How do we find the Nash equilibrium?
  - We now need to solve the following pair of equations:

$$q_1^* = \frac{1}{2}(a - q_2^* - c)$$
$$q_2^* = \frac{1}{2}(a - q_1^* - c)$$

  - Which yields:

$$q_1^* = q_2^* = \frac{1}{3}(a - c)$$

Cournot Model

- How do we find the Nash equilibrium?
  - In fact, these are the best response functions:

$$b_1(q_2) = \frac{1}{2}(a - q_2 - c)$$
$$b_2(q_1) = \frac{1}{2}(a - q_1 - c)$$

  - The Nash equilibrium is the point where the best response functions intersect:

$$q_1^* = q_2^* = \frac{1}{3}(a - c)$$

Cournot Model

- The figure shows the two best response functions and the point (q_1^*, q_2^*) where they intersect (i.e., the Nash equilibrium)

Outline

- Exercises
- Application of a Nash Equilibrium: Duopoly
  - Cournot Model
- **Mixed Strategy**
- Exercise
- Final remarks - Mixed Strategy

Mixed Strategy

- Before we start defining mixed strategy, let us look at a game called **Matching Pennies**:
  - Two agents
  - Each agent's action set is $\{Head, Tail\}$

Mixed Strategy

- **Matching Pennies**:
  - Information about the payoffs:
    - Imagine that each agent has a penny and must choose whether to display it with heads or tails facing up
    - If the **two pennies match** (i.e., both are heads up or both are tails up) then **agent 2 wins** agent 1's penny
    - If the **pennies do not match** then **agent 1 wins** agent 2's penny

Mixed Strategy

- The payoff matrix of the **Matching Pennies**:

This is a zero-sum game!

		Agent 2	
		Heads	***Tails***
Agent 1	***Heads***	-1, 1	1, -1
	Tails	1, -1	-1, 1

What is the Nash equilibrium?

Mixed Strategy

- The payoff matrix of the **Matching Pennies**:

		Agent 2	
		Heads	***Tails***
Agent 1	***Heads***	-1, 1	1, -1
	Tails	1, -1	-1, 1

No joint action can satisfy NE…

Mixed Strategy

- The **distinguishing feature** of Matching Pennies is that **each agent would like to outguess the other**

- Versions of this game also arise in **poker, baseball,** and other settings

  - In poker, the analogous question is how often to bluff:

    - if **agent i is known never to bluff** then i's opponents will fold whenever i bids aggressively, thereby making it worthwhile for i to bluff on occasion

    - on the other hand, **bluffing too often** is *also a* losing strategy

Mixed Strategy

- In any game in which **each agent would like to outguess** the other(s):
  - **there is no Nash equilibrium** (if we use the previous definition)
  - This happens because **the solution to such a game necessarily involves uncertainty** about what the agents will do

Mixed Strategy

- We now introduce the notion of a ***mixed strategy***
  - We will interpret in terms of one agent's uncertainty about what another agent will do
- Formally:
  - **Definition**: *A* **mixed strategy** *for an agent* i *is a probability distribution over his actions* $a_i \in A_i$

Mixed Strategy

- We will hereafter refer to the actions in A_i as agents i's ***pure strategies***

- For example, in Matching Pennies:

  - A_i consists of the two pure strategies: $Heads$ and $Tails$

  - Hence, a **mixed strategy for agent $\boldsymbol{i}$** is the probability distribution $(q, 1-q)$, where:
    - q is the probability of choosing $Heads$
    - $1-q$ *is* the probability of choosing $Tails$
    - $0 \leq q \leq 1$.

  - The mixed strategy (0,1) is simply the pure strategy $Tails$; likewise, the mixed strategy (1,0) is the pure strategy $Heads$

Mixed Strategy

- Recall the previous definition of Nash equilibrium:
  - NE guarantees that **each agent's pure strategy is a best response to the other agents' pure strategies**
- To **extend** the previous definition to **include mixed strategies**:
  - we simply require that **each agent's *mixed* strategy be a best response to the other agents' *mixed* strategies**

Mixed Strategy

- This **extended definition subsumes the earlier one** because:
  - Any pure strategy can be represented as the mixed strategy that puts zero probability on all of the agent's other pure strategies
- **Computing agent i's best response** to a mixed strategy by agent j:
  - We can interpret agent j's mixed strategy as representing agent i's uncertainty about what agent j will do

Mixed Strategy

- Let us return to the **Matching Pennies**:

		Agent 2	
		Heads	***Tails***
Agent 1	***Heads***	-1, 1	1, -1
	Tails	1, -1	-1, 1

What is the mixed strategy Nash equilibrium?

Mixed Strategy

- Let us now guess that **both agents randomize**
- Suppose that agent 1 believes that agent 2 will choose
  - $Heads$ with probability q
  - $Tails$ with probability $1 - q$
- Consequently, **agent 1 believes that agent 2 will play the mixed strategy** $(\boldsymbol{q}, \mathbf{1} - \boldsymbol{q})$

Mixed Strategy

- Given this belief, **agent 1's expected payoff for choosing $\boldsymbol{Heads}$** given agent 2's mixed strategy $(q, 1-q)$:

$$EU_1(Heads) = q\,(\mathbf{-1}) + (1 - q)\,\mathbf{1} = 1 - 2q$$

And **agent 1's expected payoff for choosing $\boldsymbol{Tails}$** given agent 2's mixed strategy $(q, 1-q)$:

$$EU_1(Tails) = q\,\mathbf{1} + (1 - q)(\mathbf{-1}) = 2q - 1$$

		Agent 2	
		Heads	***Tails***
Agent 1	***Heads***	-1, 1	1, -1
	Tails	1, -1	-1, 1

Mixed Strategy

- Let us first analyze $\boldsymbol{EU_1(Heads) > EU_1(Tails)}$
  - Note that $1 - 2q > 2q - 1$ if and only if $q < 1/2$
  - Hence, **agent 1's best pure-strategy response is $\boldsymbol{Heads}$** if $q < 1/2$
- Let us now analyze $\boldsymbol{EU_1(Heads) < EU_1(Tails)}$
  - Note that $1 - 2q < 2q - 1$ if and only if $q > 1/2$
  - Hence, **agent 1's best pure-strategy response is $\boldsymbol{Tails}$** if $q > 1/2$

Mixed Strategy

- **What happens when $q = \frac{1}{2}$?**
  - **This is Agent 1's best response when Agent 2 makes him indifferent between $Heads$ or $Tails$**
    - i.e., $EU_1(Heads)$ = $EU_1(Tails)$
  - **In other words, Agent 1 will only best respond with a mixed strategy when Agent 2 makes him indifferent**

Mixed Strategy

- Suppose that agent 2 believes that agent 1 will choose
  - $Heads$ with probability r
  - $Tails$ with probability $1 - r$
- Consequently, **agent 2 believes that agent 1 will play the mixed strategy** $(\boldsymbol{r}, \mathbf{1} - \boldsymbol{r})$

Mixed Strategy

- Given this belief, **agent 2's expected payoff for choosing $\boldsymbol{Heads}$** given agent 1's mixed strategy $(r, 1-r)$:

$$EU_2(Heads) = r\,1 \;+\; (1-r)\,(-1) \;= 2r - 1$$

And **agent 2's expected payoff for choosing $\boldsymbol{Tails}$** given agent 1's mixed strategy $(r, 1-r)$:

$$EU_2(Tails) = r\,(-\mathbf{1}) + \;(1-r)\mathbf{1} = 1 - 2r$$

		Agent 2	
		Heads	***Tails***
Agent 1	***Heads***	-1, 1	1, -1
	Tails	1, -1	-1, 1

Mixed Strategy

- Let us first analyze $\boldsymbol{EU_2(Heads) > EU_2(Tails)}$
  - Note that $2r - 1 > 1 - 2r$ if and only if $r > 1/2$, thus:
  - **Agent 2's best pure-strategy response is $\boldsymbol{Heads}$** if $r > 1/2$
- Let us now analyze $\boldsymbol{EU_2(Heads) < EU_2(Tails)}$
  - Note that $2r - 1 < 1 - 2r$ if and only if $r < 1/2$, thus:
  - **Agent 2's best pure-strategy response is $\boldsymbol{Tails}$** if $r < 1/2$

Mixed Strategy

- **What happens when $r = \frac{1}{2}$?**
  - **This is Agent 2's best response when Agent 1 makes him indifferent between $Heads$ or $Tails$**
    - i.e., $EU_1(Heads) = EU_1(Tails)$
  - **In other words, Agent 2 will only best respond with a mixed strategy when Agent 1 makes him indifferent**

Mixed Strategy

- **So how do we find the mixed strategy Nash equilibrium?**
  - If agent 1 best-responds with a mixed strategy, then agent 2 must make him indifferent between $Heads$ and $Tails$:

$$EU_1(Heads) = EU_1(Tails)$$
$$1 - 2q = 2q - 1$$
$$\boldsymbol{q = \frac{1}{2}}$$

  - If agent 2 best-responds with a mixed strategy, then agent 1 must make her indifferent between $Heads$ and $Tails$:

$$EU_2(Heads) = EU_2(Tails)$$
$$2r - 1 = 1 - 2r$$
$$\boldsymbol{r = \frac{1}{2}}$$

Mixed Strategy

- **So how do we find the mixed strategy Nash equilibrium?**

  - Thus, the mixed strategies $\left(\frac{1}{2}, \frac{1}{2}\right), \left(\frac{1}{2}, \frac{1}{2}\right)$ are a Nash equilibrium

Outline

- Exercises
- Application of a Nash Equilibrium: Duopoly
  - Cournot Model
- Mixed Strategy
- **Exercise**
- Final remarks - Mixed Strategy

Exercise

- **Battle of the sexes:**
  - A man and woman want to get together for an evening of entertainment, but they have no means of communication
  - They can either go to the ballet or the fight
    - The man prefers going to the fight
    - The woman prefers going to the ballet
    - But they both prefer being together than being alone

Exercise

- How many agents?
- What are the action sets?
- What are the payoffs?
- **Is there a pure strategy Nash equilibria?**
- **Is there a mixed strategy Nash equilibrium?**

Outline

- Exercises
- Application of a Nash Equilibrium: Duopoly
  - Cournot Model
- Mixed Strategy
- Exercise
- **Final remarks - Mixed Strategy**

Final remarks - Mixed Strategy

- What does it mean to play a mixed strategy?
  - Randomize to **confuse** your opponent
    - E.g., the matching pennies
      - An equilibrium only exists if we are confused about each other
  - Randomize when **uncertain** about the other's action
    - E.g., the battle of sexes

Final remarks - Mixed Strategy

- What does it mean to play a mixed strategy?
  - Mixed strategies are a concise description of what might happen in a **repeated play**
    - Count of pure strategies in the limit
  - Mixed strategies describe **population dynamics**
    - 2 agents chosen from a population, all having deterministic strategies
    - Mixed strategies gives the probability of getting each pure strategies

Thank You

rui.prada@tecnico.ulisboa.pt

Multiagent decision making and Coordination

Outline

- **Introduction to multiagent coordination**
- Coordination games
- Social conventions
- Social conventions with communication
- Coordination games + Social Conventions (with 3 agents)
- Roles
- Roles with communication
- Coordination Graphs

Multiagent Coordination

- **What is coordination in multiagent systems?**
  - *Coordination* is managing the interdependencies between activities
  - For example:
    - A non-sharable resource in the environment
    - Agents need to coordinate to use this resource

Multiagent Coordination

- **What is coordination in multiagent systems?**
  - *Coordination* is managing **robustly and efficiently** the interdependencies between activities
    - For example:
      - Two teammate soccer robots must coordinate their actions when deciding who should go for the ball

Multiagent Coordination

- **Multiagent coordination** is relevant for and **multiagent cooperation**
  - *Cooperation* is working together as a **team** to achieve a **shared goal**
  - **But we should use coordination mechanisms** within teamwork settings so that teams can efficiently achieve their goals

Multiagent Coordination

- For example, in collaborative/cooperative agents, coordination mechanisms ensure that:
  - Agents **do not obstruct** each other when taking actions
  - Agents **efficiently** perform joint actions
  - These actions serve the **common goal of the team**

Multiagent Coordination

- **Informally:**
  - **Coordination** can be regarded as the process where every agent's individual decision leads to **a good joint action for the group**

Outline

- Introduction to multiagent coordination
- **Coordination games**
- Social conventions
- Social conventions with communication
- Coordination games + Social Conventions (with 3 agents)
- Roles
- Roles with communication
- Coordination Graphs

Coordination Games

- We can model a coordination problem as a **coordination game** using the tools from game theory
  - We can use the **normal-form representation**
    - Actions sets and payoffs
  - Solve it using some **solution concept**
    - Nash equilibrium

Coordination Games

- Consider the **stag hunt game** with two hunters:
  - If they both hunt hares, they each capture half of the hares in the range

  - If one hunts the stag and the other hunts hares, the stag hunter goes home empty-handed while the hare hunter captures all the hares

  - Finally, if both hunt the stag, then each of their shares of the stag is greater than the value of all the hares

Coordination Games

- The payoff matrix of the **stag hunt game**:

		Hunter 2	
		Stag	***Hare***
Hunter 1	***Stag***	3, 3	0, 2
	Hare	2, 0	1, 1

Coordination Games

- What is the NE in the **stag hunt game?**

		Hunter 2	
		Stag	***Hare***
Hunter 1	***Stag***	3, 3	0, 2
	Hare	2, 0	1, 1

Nash equilibria

Which equilibrium should the hunters choose?

Coordination Games

- **No coordination** in the stag hunt game:
  - The action to hunt a ***Stag*** is a **risk-taking strategy**
    - If **both hunters choose to hunt the *Stag***, then both get the **highest payoff**
    - If **only one hunter chooses to hunt the *Stag*** then he gets the **lowest payoff**
  - The action to hunt a ***Hare*** is a **conservative strategy**
    - The hunter will **never get the highest payoff or lowest payoff**
    - The hunter will **never go home empty-handed**

Coordination Games

- **With coordination** in the stag hunt game:
  - What if both hunters enter into some kind of **agreement** (in advance) and both **trust** each other?
    - One agreement could be to choose the joint action that is **Pareto optimal** (or strictly Pareto efficient)

Coordination Games

- **Definition:** *A joint action* a **Pareto dominates** *joint action* a' *if for all* $i \in N$, $u_i(a) \geq u_i(a')$, *and there exists some* $j \in N$ *for which* $u_j(a) > u_j(a')$

- **Definition:** *A joint action* a *is* **Pareto optimal**, *or* strictly Pareto efficient, *if there does not exist another joint action* $a' \in A$ *that Pareto dominates* a

Coordination Games

- What is the NE in the **stag hunt game?**

Hunter 2

Hunter 1		***Stag***	***Hare***
	Stag	3, 3	0, 2
	Hare	2, 0	1, 1

Nash equilibria

Pareto optimal

Coordination Games

- We can now formally define **coordination**:
  - ***The process in which a group of agents choose a single Pareto optimal Nash equilibrium in a game***

Coordination Games

- Example of two autonomous vehicles at a crossroad

Coordination Games

- Example:
  - Each agent wants to cross first
  - But if they both cross they will crash
- **How can we represent this game with the normal-form representation and solve it?**

Coordination Games

- The payoff matrix of the **car coordination game**:

		Car 2	
		Cross	***Stop***
Car 1	***Cross***	-1, -1	1, 0
	Stop	0, 1	0, 0

Coordination Games

- How do we solve this coordination game?
  - Pareto optimal Nash equilibria

		Car 2	
		Cross	***Stop***
Car 1	***Cross***	-1, -1	1, 0
	Stop	0, 1	0, 0

Which equilibrium should the agents choose?

Coordination Games

Coordination Games

- Robot soccer example (within a **teamwork** or **collaborative** setting):
  - Two robots (of the same team) want to run to get the ball
  - But if they both try to get the ball at the same time, they will crash
- **How can we represent this game with the normal-form representation and solve it?**

Coordination Games

		Robot 2	
		Run	***Stay***
Robot 1	***Run***	-1, -1	1, 1
	Stay	1, 1	0, 0

- In the case of n **collaborative agents**, all agents in the team share the same payoff function:

$$u_1(a) = \cdots = u_n(a) \equiv u(a)$$

- These game are known as **pure coordination games** or **team games**

Coordination Games

- How do we solve this coordination game?
  - Pareto optimal Nash equilibria

		Robot 2	
		Run	***Stay***
Robot 1	***Run***	-1, -1	1, 1
	Stay	1, 1	0, 0

Which equilibrium should the agents choose?

Outline

- Introduction to multiagent coordination
- Coordination games
- **Social conventions**
- Social conventions with communication
- Coordination games + Social Conventions (with 3 agents)
- Roles
- Roles with communication
- Coordination Graphs

Social Conventions

- To solve a coordination problem:
  - A group of agents are faced with the problem of **how to choose their actions** in order to select the **same Nash equilibrium**
  - Unfortunately, **there is not a single recipe** that can be used for all games
  - Nevertheless, we can **devise recipes that will instruct the agents** on how to choose a single equilibrium for a given game

Social Conventions

- A **social convention** (or **social law**) is such a recipe
  - It places **constraints on the possible action choices** of the agents
  - It can be regarded as a **rule that dictates how the agents should choose their actions in a coordination game** in order to reach an equilibrium
  - Moreover, **given that the convention has been established** and is common knowledge among agents, **no agent can benefit from not abiding by it**

Social Conventions

- **Boutilier (1996)** proposed a **general convention** that achieves coordination in a large class of systems and **is very easy to implement:**
  - The convention **assumes a unique ordering scheme** of joint actions that is common knowledge among agents
  - In a particular game, **each agent first computes all equilibria** of the game, and **then selects the first equilibrium according to this ordering scheme**

Boutilier, C. (1996). Planning, learning and coordination in multiagent decision processes.
In *Proc. Conf. on Theoretical Aspects of Rationality and Knowledge*, Renesse, The Netherlands.

Social Conventions

- Two agents who want to go to the movies together but not alone

		Agent 2	
		Thriller	***Comedy***
Agent 1	***Thriller***	1, 1	0, 0
	Comedy	0, 0	1, 1

Social Conventions

- **Each agent first computes all equilibria** of the game

		Agent 2	
		Thriller	***Comedy***
Agent 1	***Thriller***	1, 1	0, 0
	Comedy	0, 0	1, 1

Social Conventions

- **Then each agent selects the first equilibrium according to an ordering scheme:**
  - **Order the agents** by $1 \succ 2$
    - meaning that agent 1 has ‘priority’ over agent 2
  - **Order the actions** by *Thriller* $\succ$ *Comedy*
- ***What action should each agent choose?***

Social Conventions

- **Following the ordering scheme:**

		Agent 2	
		Thriller	***Comedy***
Agent 1	***Thriller***	1, 1	0, 0
	Comedy	0, 0	1, 1

Social Conventions

- Hence, the first equilibrium in the resulting ordering of joint actions is:

  (*Thriller*, *Thriller*)

- And this will be the **unanimous choice of the agents**

- **Now each agent can choose his individual action**

Social Conventions

- Recall the **car coordination game**:

		Car 2	
		Cross	***Stop***
Car 1	***Cross***	-1, -1	1, 0
	Stop	0, 1	0, 0

Social Conventions

- **Each agent first computes all equilibria** of the game

		Car 2	
		Cross	***Stop***
Car 1	***Cross***	-1, -1	1, 0
	Stop	0, 1	0, 0

Social Conventions

- **Then each agent selects the first equilibrium according to an ordering scheme:**
  - **The driver coming from the right will always have priority over the other driver**
    - Note that agents need to have **more information about the environment and/or other agents** (and not just the payoffs and action sets)
    - Let us assume that Car 1 sees Car 2 coming from the right
      - Then Car 2 has 'priority' over Car 1
  - **Order the actions** by *Cross* $\succ$ *Stop*
- ***What action should each agent choose?***

Social Conventions

- **Following the ordering scheme:**

		Car 2	
		Cross	***Stop***
Car 1	***Cross***	-1, -1	1, 0
	Stop	0, 1	0, 0

Social Conventions

- Hence, the first equilibrium in the resulting ordering of joint actions is:

  (*Stop*, *Cross*)

- And this will be the **unanimous choice of the agents**

- **Now each agent can choose his individual action**

Exercise

- **Battle of the sexes:**
  - A man and woman want to get together for an evening of entertainment, but they have no means of communication
  - They can either go to the ballet or the fight
    - The man prefers going to the fight
    - The woman prefers going to the ballet
    - But they both prefer being together than being alone
- **Create an ordering scheme and solve the coordination problem**

Outline

- Introduction to multiagent coordination
- Coordination games
- Social conventions
- **Social conventions with communication**
- Coordination games + Social Conventions (with 3 agents)
- Roles
- Roles with communication
- Coordination Graphs

Social Conventions with Communication

- When **communication** is available:
  - **We only need to impose an ordering of the agents**
    - And this ordering is **common knowledge**

Social Conventions with Communication

- We first assume an ordering $i = 1, \dots, n$ of agents

- Coordination can now be achieved by the following algorithm:

  1. Each agent i (except agent 1) waits until all previous agents $1, \dots, i-1$ in the ordering have broadcast their chosen actions

  2. agent i computes its component a_i^* of an equilibrium that is consistent with the choices of the previous agents

  3. agent i broadcasts a_i^* to all agents that have not chosen an action yet

Social Conventions with Communication

- Note that:
  - The **fixed ordering of the agents** together with the **wait/send primitives** induce a **synchronized sequential execution order of the coordination algorithm**

Social Conventions with Communication

- Recall the **car coordination game**:

		Car 2	
		Cross	***Stop***
Car 1	***Cross***	-1, -1	1, 0
	Stop	0, 1	0, 0

Social Conventions with Communication

- Let us assume the following ordering of agents:

(Car 1, Car2)

- Car 1 chooses action a_1^*= *Cross* of an equilibrium

		Car 2	
		Cross	***Stop***
Car 1	***Cross***	-1, -1	<u>1</u>, <u>0</u>
	Stop	<u>0</u>, <u>1</u>	0, 0

- Car 1 broadcasts a_1^* = *Cross* to all agents that have not chosen an action yet
  - i.e., sends a message to Car 2

Social Conventions with Communication

- Car 2 waits until all previous agents in the ordering have broadcast their chosen actions
  - Thus, it waits a message from Car 1: a_1^* = *Cross*

- Car 2 computes its component a_2^* of an equilibrium that is consistent with the choices of the previous agents
  - Thus, a_2^* = *Stop*

		Car 2	
		Cross	***Stop***
Car 1	***Cross***	-1, -1	1, 0
	~~***Stop***~~	~~0, 1~~	~~0, 0~~

Outline

- Introduction to multiagent coordination
- Coordination games
- Social conventions
- Social conventions with communication
- **Coordination games + Social Conventions (with 3 agents)**
- Roles
- Roles with communication
- Coordination Graphs

Coordination Games

- Example of **THREE** autonomous vehicles at a crossroad

Coordination Games

Coordination Games

- Example:
  - Each agent/car wants to cross first
  - But if they two cross they will crash
- **How can we represent this game with the normal-form representation and solve it?**

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, stop, stop)	1	0	0
(cross, cross, stop)	-1	-1	0
(cross, stop, cross)	-1	0	-1
(cross, cross, cross)	-1	-1	-1
(stop, stop, stop)	0	0	0
(stop, cross, stop)	0	1	0
(stop, stop, cross)	0	0	1
(stop, cross, cross)	0	-1	-1

Coordination Games

- Let us calculate the best response functions:

  - $B_1(a_2 = stop, a_3 = stop) = ?$
  - $B_1(a_2 = cross, a_3 = stop) = ?$
  - $B_1(a_2 = stop, a_3 = cross) = ?$
  - $B_1(a_2 = cross\ a_3 = cross) = ?$

  - $B_2(a_1 = stop, a_3 = stop) = ?$
  - $B_2(a_1 = cross, a_3 = stop) = ?$
  - $B_2(a_1 = stop, a_3 = cross) = ?$
  - $B_2(a_1 = cross, a_3 = cross) = ?$

  - $B_3(a_1 = stop, a_2 = stop) = ?$
  - $B_3(a_1 = cross, a_2 = stop) = ?$
  - $B_3(a_1 = stop, a_2 = cross) = ?$
  - $B_3(a_1 = cross\ a_2 = cross) = ?$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, stop, stop)	1	0	0
(stop, stop, stop)	0	0	0

	$stop, stop$
cross	1, 0, 0
stop	0, 0, 0

$$B_1(a_2 = stop, a_3 = stop) = \ cross$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, cross, stop)	-1	-1	0
(<u>stop</u>, cross, stop)	0	1	0

	$cross, stop$
cross	-1, -1, 0
<u>stop</u>	0, 1, 0

$$B_1(a_2 = cross, a_3 = stop) = stop$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, stop, cross)	-1	0	-1
(<u>stop</u>, stop, cross)	0	0	1

	$stop, cross$
cross	-1, 0, -1
<u>stop</u>	0, 0, 1

$$B_1(a_2 = stop, a_3 = cross) = stop$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, cross, cross)	-1	-1	-1
(<u>stop</u>, cross, cross)	0	-1	-1

	$cross, cross$
cross	-1, -1, -1
<u>stop</u>	0, -1, 1

$$B_1(a_2 = cross, a_3 = cross) = stop$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, stop, stop)	1	0	0
(cross, cross, stop)	-1	-1	0
(cross, stop, cross)	-1	0	-1
(cross, cross, cross)	-1	-1	-1
(stop, stop, stop)	0	0	0
(stop, cross, stop)	0	1	0
(stop, stop, cross)	0	0	1
(stop, cross, cross)	0	-1	-1

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(stop, stop, stop)	0	0	0
(<u>stop</u>, <u>cross</u>, stop)	0	1	0

	$stop, stop$
stop	0, 0, 0
<u>cross</u>	0, 1, 0

$$B_2(a_1 = stop, a_3 = stop) = cross$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, stop, stop)	1	0	0
(cross, cross, stop)	-1	-1	0

	$cross, stop$
stop	1, 0, 0
cross	-1, -1, 0

$$B_2(a_1 = cross, a_3 = stop) = stop$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(stop, stop, cross)	0	0	1
(stop, cross, cross)	0	-1	-1

	$stop, cross$
stop	0, 0, 1
cross	0, -1, -1

$$B_2(a_1 = stop, a_3 = cross) = stop$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, <u>stop</u>, cross)	-1	0	-1
(cross, cross, cross)	-1	-1	-1

	$cross, cross$
<u>stop</u>	-1, 0, -1
cross	-1, -1, -1

$$B_2(a_1 = cross, a_3 = cross) = stop$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, stop, stop)	1	0	0
(cross, cross, stop)	-1	-1	0
(cross, stop, cross)	-1	0	-1
(cross, cross, cross)	-1	-1	-1
(stop, stop, stop)	0	0	0
(stop, cross, stop)	0	1	0
(stop, stop, cross)	0	0	1
(stop, cross, cross)	0	-1	-1

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(stop, stop, stop)	0	0	0
(<u>stop</u>, <u>stop</u>, <u>cross</u>)	0	0	1

	$stop, stop$
stop	0, 0, 0
<u>cross</u>	0, 0, 1

$$B_3(a_1 = stop, a_2 = stop) = cross$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, stop, stop)	1	0	0
(cross, stop, cross)	-1	0	-1

	$cross, stop$
stop	1, 0, 0
cross	-1, 0, -1

$$B_3(a_1 = cross, a_2 = stop) = stop$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(stop, cross, stop)	0	1	0
(stop, cross, cross)	0	-1	-1

	$stop, cross$
stop	0, 1, 0
cross	0, -1, -1

$$B_3(a_1 = stop, a_2 = cross) = stop$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, cross, <u>stop</u>)	-1	-1	0
(cross, cross, cross)	-1	-1	-1

	$cross, cross$
<u>stop</u>	-1, -1, 0
cross	-1, -1, -1

$$B_3(a_1 = cross, a_2 = cross) = stop$$

Coordination Games

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, stop, stop)	1	0	0
(cross, cross, stop)	-1	-1	0
(cross, stop, cross)	-1	0	-1
(cross, cross, cross)	-1	-1	-1
(stop, stop, stop)	0	0	0
(stop, cross, stop)	0	1	0
(stop, stop, cross)	0	0	1
(stop, cross, cross)	0	-1	-1

Pareto optimal Nash equilibria

Coordination Games & Social Conventions

- **Then each agent selects the first equilibrium according to an ordering scheme:**
  - **The driver coming from the right will always have priority over the other driver**
  - **Order the actions** by *Cross* $\succ$ *Stop*
- ***What action should each agent choose?***

Coordination Games & Social Conventions

- Car 1 sees Car 2 and Car 3 coming from the right
- Car 2 sees Car 3 coming from the right

- Hence, Car 3 $\succ$ Car 2 $\succ$ Car 1

Coordination Games & Social Conventions

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(cross, stop, stop)	1	0	0
(cross, cross, stop)	-1	-1	0
(cross, stop, cross)	-1	0	-1
(cross, cross, cross)	-1	-1	-1
(stop, stop, stop)	0	0	0
(stop, cross, stop)	0	1	0
(stop, stop, cross)	0	0	1
(stop, cross, cross)	0	-1	-1

Now each agent can choose his individual action

Outline

- Introduction to multiagent coordination
- Coordination games
- Social conventions
- Social conventions with communication
- Coordination games + Social Conventions (with 3 agents)
- **Roles**
- Roles with communication
- Coordinaiton Graphs

Roles

- Coordination by **social conventions** relies on the following assumption:
  - An **agent can compute all equilibria in a game before choosing a single one**
- However, **computing equilibria can be expensive** when the agents' action sets are large

Roles

- Hence, it makes sense **to start trying to reduce the size of the action sets**
- Such a reduction can lead to:
  - computational advantages in terms of **speed**
  - simplify the **equilibrium selection problem**

Roles

- A **natural way to reduce the agents' action sets** is to assign **roles** to the agents
- Formally:
  - **Given a particular state,** a role can be viewed as a **masking operator on the agent's action set**

Roles

- In practical terms:
  - if an **agent is assigned a role** at a particular state, then some of the **agent's actions are deactivated** at this state

For instance, a **robot soccer agent** that is currently with a **defender role cannot attempt to *Score***

Roles

- **Example**: two agents who want to go to the movies together but not alone

- if **Agent 2** is assigned a role that **forbids him to select the action *Thriller***
  - Because **Agent 2 is under 12**

		Agent 2	
		Thriller	***Comedy***
Agent 1	***Thriller***	1, 1	0, 0
	Comedy	0, 0	1, 1

Roles

So how can we assign a role to an agent?

- Suppose that:
  - there are n available roles (not necessarily distinct)
  - the state is fully observable to the agents

Roles

- Suppose that the following **facts are common knowledge** among agents:
  - There is a **fixed ordering** $\{1,2,\ldots,n\}$ **of the roles**
    - In other words, role 1 must be assigned first, followed by role 2, etc.

Roles

- Suppose that the **following facts are common knowledge** among agents:
  - For each role there is a **function that assigns to each agent a 'potential'**
    - The potential reflects **how appropriate that agent is for the specific role**, given the current state
  - For example, in robot soccer scenario:
    - the **potential** of a robot for the **role attacker** can be given by its **negative Euclidean distance to the ball**

Roles

- Suppose that the **following facts are common knowledge** among agents:
  - Each agent can be assigned only one role

Roles

- Then **role assignment** can be carried out, for instance, by a **greedy algorithm**:
  - each role (starting from role 1) is assigned to the agent that has the highest potential for that role
  - and so on until all agents have been assigned a role

Roles

- Example: Robot soccer

Roles

- 3 robots: 1, 2, and 3
- Action sets: $A_1, A_2, A_3 = \{run, stay\}$

Roles

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(run, run, run)	-1	-1	-1
(run, run, stay)	-1	-1	-1
(run, stay, run)	-1	-1	-1
(run, stay, stay)	1	1	1
(stay, run, run)	-1	-1	-1
(stay, run, stay)	1	1	1
(stay, stay, run)	1	1	1
(stay, stay, stay)	0	0	0

Pareto optimal Nash equilibria

Roles

- Fixed ordering of the roles = {Attacker, Goalkeeper, Defender}
- **Potential** for role **Attacker**: negative Euclidean distance to the ball
- **Potential** for role **Goalkeeper**: negative Euclidean distance to its own team's goal
- **Potential** for role **Defender**: positive Euclidean distance to the ball

Roles

- Assigning the **Attacker** role (**A**)
  - Potential is the negative Euclidean distance to the ball

Roles

- Assigning the **Goalkeeper** role (**G**)
  - Potential is the negative Euclidean distance to its own team's goal

Roles

- Assigning the **Defender** role (**D**)
  - Potential is the positive Euclidean distance to the ball

Roles

- Hence, if an **agent is assigned a role** at a particular state then some of the **agent's actions are deactivated** at this state
  - The role **Defender** and **Goalkeeper** cannot choose the action $\boldsymbol{run}$ (so they can only choose $\boldsymbol{stay}$)
  - The role **Attacker** can either choose the action $\boldsymbol{run}$ or $\boldsymbol{stay}$

Joint actions (a)	$u_1(a)$	$u_2(a)$	$u_3(a)$
(stay, stay, run)	1	1	1
(stay, stay, stay)	0	0	0

Outline

- Introduction to multiagent coordination
- Coordination games
- Social conventions
- Social conventions with communication
- Coordination games + Social Conventions (with 3 agents)
- Roles
- **Roles with communication**
- Coordination Graphs

Roles with communication

- When **communication is not available:**
  - **Each agent can run** the previous algorithm identically and **in parallel**
    - Assuming that each agent can compute the **potential of each other agent**

Roles with communication

- When **communication is available:**
  - An agent only needs to compute its own **potentials** for the set of roles
  - Then broadcast them to the rest of the agents.
  - Next it can wait for all other potentials to arrive in order to compute the assignment of roles to agents as the previous algorithm

Roles with communication

<u>For</u> each agent i in parallel
    $I = \{\}$.
    <u>For</u> each role $j = 1, \ldots, n$
        Compute the potential r_{ij} of agent i for role j.
        Broadcast r_{ij} to all agents.
    <u>End</u>
    Wait until all $r_{i'j}$, for $j = 1, \ldots, n$, are received.
    <u>For</u> each role $j = 1, \ldots, n$
        Assign role j to agent $i^* = \arg\max_{i' \notin I} \{r_{i'j}\}$.
        Add i^* to I.
    <u>End</u>
<u>End</u>

Final remarks

- In the **communication-free algorithm**
  - The greedy algorithm has a complexity of $O(n^2)$
- In the **communication-based algorithm**
  - Each agent needs to compute $O(n)$ potentials (its own)
  - However, this is added to the total number $O(n^2)$ of potentials that need to be broadcast and processed by the agents

Exercise

Cleaning robot coordination problem

Exercise

- 3 cleaning robots: 1, 2, and 3
- Action sets: $A_1, A_2, A_3 = \{clean, charge, stay\}$
- Ordering of roles = {Need Charging (NC), Go Clean (GC), Wait for Work (WfW)}
- Potential for NC: negative of the battery level
- Potential for GC: negative distance to dirt
- Potential for WfW: positive distance to dirt

Exercise

Current State

Exercise

- Deactivated actions for each role:
  - Need Charging (NC): cannot clean and cannot stay
  - Go Clean (GC): cannot stay and cannot charge
  - Wait for Work (WfW): cannot clean
- **Use the communication-based algorithm to assign the roles and solve the coordination problem**

Exercise

- Role assignment
  - Need Charging (NC): 3
  - Go Clean (GC): 1
  - Wait for Work (WfW): 3
- **Can we improve this?**

Exercise

- **Change the ordering?**
- **Role admissibility / activation conditions**
  - Add a function that tests if the role is needed in the current state

```
NC.available(A)
        If A.battery_level < 15%
```

Outline

- Introduction to multiagent coordination
- Coordination games
- Social conventions
- Social conventions with communication
- Coordination games + Social Conventions (with 3 agents)
- Roles
- Roles with communication
- **Coordination Graphs**

Roles

- **Roles** can help us solve coordination game by **reducing the agents' action sets** prior to **computing the equilibria**

- However, computing equilibria in a subgame can still be a **difficult task** when the **number of agents is large**

- Recall that the number of joint actions is exponential in the number of agents:

$$m^n$$

where m is the number of the agents' actions and n is the number of agents

Coordination Graphs

- We also **need a method** that **reduces the number of agents** in a coordination game
- **Coordination graph** (Guestrin et al., 2002) is a framework for **solving large-scale coordination problems**

Coordination Graphs

- A coordination graph **decomposes a coordination game** into several smaller **subgames that are easier to solve**
  - In **roles**, we reduce the action sets and form a **single subgame**
  - In **coordination graphs**, we form **various subgames**, each involving a smaller number of agents

Coordination Graphs

- **Main assumption** in coordination graphs:
  - the **global payoff function** $u(a)$ can be written **as a linear combination of $\boldsymbol{k}$ local payoff functions** f_j, for $j = 1, \ldots, k$, each involving fewer agents
- For instance, a **coordination problem** with $\boldsymbol{n = 4}$ **agents** and $\boldsymbol{k = 3}$ **local payoff functions**, each involving two agents, could have the following decomposition:

$$u(a) = f_1(a_1, a_2) + f_2(a_1, a_3) + f_3(a_3, a_4)$$

Coordination Graphs

- Example of the decomposition of the global payoff function:

$$u(a) = f_1(a_1, a_2) + f_2(a_1, a_3) + f_3(a_3, a_4)$$

- Note:
  - $f_1(a_1, a_2)$ involves only agents 1 and 2, with their actions a_1 and a_2
  - $f_2(a_1, a_3)$ involves only agents 1 and 3, with their actions a_1 and a_3
  - $f_3(a_3, a_4)$ involves only agents 3 and 4, with their actions a_3 and a_4

Coordination Graphs

- Example of the decomposition of the global payoff function:

$$u(a) = f_1(a_1, a_2) + f_2(a_1, a_3) + f_3(a_3, a_4)$$

- The decomposition can be represented with the following graph:

Coordination Graphs

- **Practical examples** that could use coordination graphs:
  - **Servers in a computer network** that need to coordinate in order to **optimize the overall network traffic**

Coordination Graphs

- **Practical examples** that could use coordination graphs:
  - Within a corporation, **offices in nearby cities** may need to coordinate in order to **maximize global sales**

Coordination Graphs

- **Practical examples** that could use coordination graphs:
  - In a soccer team, **nearby players** may need to coordinate in order to **improve team performance**

Coordination Graphs

- Linear coordination

Coordination Graphs

- Cycle/ring coordination

Coordination Graphs

- Grid coordination

Coordination Graphs

- Example: Robot soccer

Coordination Graphs

- 5 robots: 1, 2, 3, 4 and 5

- Action sets: $A_1, A_2, A_3, A_4, A_5 = \{stay, pass\}$

Coordination Graphs

$$u(a) = f_1(a_1, a_3) + f_2(a_1, a_2) + f_3(a_3, a_4) + f_4(a_4, a_5)$$

Thank You

rui.prada@tecnico.ulisboa.pt

TÉCNICO LISBOA

Multiagent decision making and Games in Extensive Form

Outline

- **Perfect-information games in extensive form**
- Strategies and equilibria
- Subgame-perfect equilibrium
- Backward induction
- Example

Extensive-form games

- Normal-form games do not incorporate any notion of **sequence or time**
  - Normal-form games assume that agents **select their actions simultaneously**
  - In many examples of normal-form games, we have considered **one-shot games**

Extensive-form games

- Extensive-form games:
  - Also known as **tree-form games**
  - An **alternative representation** that makes the **temporal structure explicit**
  - We now present the ***perfect-information*** **extensive-form games** (finite games)

Extensive-form games

- What are perfect-information games in extensive form?
  - A **tree** in the sense of **graph theory**
  - Each **node** represents the **choice** of an agent
  - Each **edge** represents an **action** of an agent
  - The **leaves** represent a **final outcome (payoffs/utility)**

Extensive-form games

- Example: **the Sharing game**
  - Imagine a **brother and sister** have to **decide how to share two indivisible and identical presents** from their parents in the following way:
    - **First the brother suggests a split**, which can be one of three—he keeps both, she keeps both, or they each keep one.
    - Then the **sister chooses whether to accept or reject the split**.

Extensive-form games

- Example: **the Sharing game**
  - If she accepts, they each get their allocated present(s), and otherwise, neither gets any gift.
  - Assume both siblings value the two presents equally and additively

Extensive-form games

The Sharing Game

1-brother
2-sister

Extensive-form games

- **Definition (Perfect-information game):** *A (finite) perfect-information game (in extensive form) is a tuple* $G = (N, A, H, Z, \chi, \rho, \sigma, u)$*, where:*
  - N *is a* ***set of*** $\boldsymbol{n}$ ***agents****;*
  - A *is a (single)* ***set of actions****;*
  - H *is a set of* ***nonterminal choice nodes****;*
  - Z *is a set of* ***terminal nodes****, disjoint from* H*;*

Extensive-form games

- **Definition (Perfect-information game):** *A (finite) perfect-information game (in extensive form) is a tuple* $G = (N, A, H, Z, \chi, \rho, \sigma, u)$*, where:*

  - $\chi: H \longmapsto 2^A$ *is the* ***action function****, which assigns to each choice node a set of possible actions;*

  - $\rho: H \longmapsto N$ *is the* ***player function****, which assigns to each nonterminal node a player (agent)* $i \in N$ *who chooses an action at that node;*

Extensive-form games

- **Definition (Perfect-information game):** *A (finite) perfect-information game (in extensive form) is a tuple* $G = (N, A, H, Z, \chi, \rho, \sigma, u)$*, where:*

  - $\sigma: H \times A \longmapsto H \cup Z$ *is the* ***successor function****, which maps a choice node and an action to a new choice node or terminal node, such that for all* $h_1, h_2 \in H$ *and* $a_1, a_2 \in A$*, if* $\sigma(h_1, a_1) = \sigma(h_2, a_2)$ *then* $h_1 = h_2$ *and* $a_1 = a_2$*; and*

  - $u = (u_1, \dots, u_n)$*, where* $u_i: Z \longmapsto \mathbb{R}$ *is a* ***real-valued utility function*** *for player (agent)* i *on the terminal nodes* Z*.*

Outline

- Perfect-information games in extensive form
- **Strategies and equilibria**
- Subgame-perfect equilibrium
- Backward induction
- Example

Strategies and Equilibria

- A **pure strategy** for an agent in a perfect-information game is:
  - A **complete specification of which deterministic action to take at every node** belonging to that agent

Strategies and Equilibria

- **Definition (Pure strategies):** *Let* $G = (N, A, H, Z, \chi, \rho, \sigma, u)$ *be a perfect-information extensive-form game. Then the pure strategies of agent* i *consist of the Cartesian product*

$$\Pi_{h \in H, \rho(h)=i}\, \chi(h).$$

Action function

Strategies and Equilibria

The Sharing game

Pure Strategies:

S_1 = {2-0, 1-1, 0-2}

S_2 = {(*yes,yes,yes*), (*yes,yes,no*), (*yes,no,yes*), (*yes,no,no*), (*no,yes,yes*), (*no, yes, no*), (*no, no, yes*), (*no, no, no*)}

Strategies and Equilibria

Let us now consider another game

Strategies and Equilibria

Pure Strategies:

$S_1 = \{(A,G), (A,H), (B,G), (B,H)\}$

$S_2 = \{(C,E), (C,F), (D,E), (D,F)\}$

Strategies and Equilibria

- **So how do we compute the Nash equilibria?**
  - The definition of best response is the same as we've seen so far!
  - The definition of Nash equilibria is the same too!

Strategies and Equilibria

- We convert a perfect-information game to an equivalent normal-form game

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)				
(A,H)				
(B,G)				
(B,H)				

Strategies and Equilibria

- We convert a perfect-information game to an equivalent normal-form game

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8			
(A,H)				
(B,G)				
(B,H)				

Strategies and Equilibria

- We convert a perfect-information game to an equivalent normal-form game

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8		
(A,H)				
(B,G)				
(B,H)				

Strategies and Equilibria

- We convert a perfect-information game to an equivalent normal-form game

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	
(A,H)				
(B,G)				
(B,H)				

Strategies and Equilibria

- We convert a perfect-information game to an equivalent normal-form game

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	8, 3
(A,H)				
(B,G)				
(B,H)				

Strategies and Equilibria

- We convert a perfect-information game to an equivalent normal-form game

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	8, 3
(A,H)	3, 8	3, 8	8, 3	8, 3
(B,G)	5, 5	2, 10	5, 5	2, 10
(B,H)	5, 5	1, 0	5, 5	1, 0

Strategies and Equilibria

- The **temporal structure** of the extensive-form game can generate a certain **redundancy** within the normal-form game

- In the previous example:

  - Normal form – 16 outcomes

  - Extensive form – 5 outcomes

Strategies and Equilibria

- **Theorem**: *Every (finite) perfect-information game in extensive form has a pure-strategy Nash equilibrium.*

- Intuition:
  - Since agents take turns, and everyone gets to see everything that happened thus far before making a move, it is never necessary to introduce randomness into action selection in order to find an equilibrium.

Strategies and Equilibria

- Find the Nash equilibria?

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	8, 3
(A,H)	3, 8	3, 8	8, 3	8, 3
(B,G)	5, 5	2, 10	5, 5	2, 10
(B,H)	5, 5	1, 0	5, 5	1, 0

Strategies and Equilibria

- We convert a perfect-information game to an equivalent normal-form game

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, <u>8</u>	<u>3</u>, <u>8</u>	<u>8</u>, 3	<u>8</u>, 3
(A,H)	3, <u>8</u>	<u>3</u>, <u>8</u>	<u>8</u>, 3	<u>8</u>, 3
(B,G)	<u>5</u>, 5	2, <u>10</u>	5, 5	2, <u>10</u>
(B,H)	<u>5</u>, <u>5</u>	1, 0	5, <u>5</u>	1, 0

Nash equilibria in this game:
{(*A, G*), (*C, F*)}, {(*A, H*), (*C, F*)}, and {(*B, H*), (*C, E*)}

Exercise

- A market has a single firm (monopoly) with $2M in cash
- A new firm is deciding whether to enter (or not) the market
  - The new firm has $2M to start the operation
- If the new firm enters, the monopolist may accept the decision or declare a price war
  - If the monopolist accepts, the new firm makes a $1M profit and the monopolist loses $1M
  - A price war is unprofitable for both firms (i.e., they lose all they have)

Exercise

- Model this game as a perfect-information extensive-form game? Present the game tree.
- Convert a perfect-information game to an equivalent normal-form game
- Find the Nash equilibria
- Analyze the equilibria and see if there is anything strange

Outline

- Perfect-information games in extensive form
- Strategies and equilibria
- **Subgame-perfect equilibrium**
- Backward induction
- Example

Subgame-Perfect Equilibrium

- Examining the converted normal-form game of an extensive-form game **obscures the game's temporal nature**

- Let us see this with our previous example
  - Let us focus on the equilibria {(*A, G*), (*C, F*)} and {(*B, H*), (*C, E*)}

Subgame-Perfect Equilibrium

- Agent 2 has **no incentive to deviate** by **changing *C* to *D***!

This is intuitive!

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, <u>8</u>	<u>3</u>, <u>8</u>	<u>8</u>, 3	<u>8</u>, 3
(A,H)	3, <u>8</u>	<u>3</u>, <u>8</u>	<u>8</u>, 3	<u>8</u>, 3
(B,G)	<u>5</u>, 5	2, <u>10</u>	5, 5	2, <u>10</u>
(B,H)	<u>5</u>, <u>5</u>	1, 0	5, <u>5</u>	1, 0

Subgame-Perfect Equilibrium

- Agent 1 has **no incentive to deviate** by **changing *A* to *B***!

This is intuitive!

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	8, 3
(A,H)	3, 8	3, 8	8, 3	8, 3
(B,G)	5, 5	2, 10	5, 5	2, 10
(B,H)	5, 5	1, 0	5, 5	1, 0

Subgame-Perfect Equilibrium

- This is also a Nash equilibrium. However…

	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	8, 3
(A,H)	3, 8	3, 8	8, 3	8, 3
(B,G)	5, 5	2, 10	5, 5	2, 10
(B,H)	5, 5	1, 0	5, 5	1, 0

Subgame-Perfect Equilibrium

- This is also a Nash equilibrium. However…

Agent 2 chooses action *E* because he knows that agent 1 would choose *H* afterwards

The behavior of agent 1 is called a *threat*

Subgame-Perfect Equilibrium

- This is also a Nash equilibrium. However…

What if agent 2 does not consider agent 1's *threat* to be credible?

If agent 2 played *F*, would agent 1 really follow through on his threat? NO

Subgame-Perfect Equilibrium

- In other words, there is something *intuitively wrong* with the equilibrium (*B,H*),(*C,E*)

- To formally capture the reason, let us define the notion of subgame.

Subgame-Perfect Equilibrium

- **Definition (Subgame of *G* rooted at *h*)**: *Given a perfect-information extensive-form game G, the subgame of G rooted at node h is the restriction of G to the descendants of h.*

- **Definition (Subgames of *G*):** *The set of subgames of G consists of all of subgames of G rooted at some node in G.*

Subgame-Perfect Equilibrium

- For example

Subgame-Perfect Equilibrium

- For example

Subgame-Perfect Equilibrium

- For example

Subgame-Perfect Equilibrium

- **Definition (Subgame-perfect equilibrium):** *The subgame-perfect equilibria (SPE) of a game G are all strategy profiles s such that for any subgame G' of G, the restriction of s to G' is a Nash equilibrium of G'.*

Subgame-Perfect Equilibrium

- Consider the NE {(*A*,*G*),(*C*,*F*)}, is it a **subgame-perfect equilibrium**?

Is this subgame a NE? YES

Subgame-Perfect Equilibrium

- Consider the NE {(*A*,*G*),(*C*,*F*)}, is it a **subgame-perfect equilibrium**?

Is this subgame a NE? YES

Subgame-Perfect Equilibrium

- Consider the NE {(*A*,*G*),(*C*,*F*)}, is it a **subgame-perfect equilibrium**?

Is this subgame a NE? YES

Subgame-Perfect Equilibrium

- Consider the NE {(*A*,*G*),(*C*,*F*)}, is it a **subgame-perfect equilibrium**?

YES

Subgame-Perfect Equilibrium

- Consider the NE {(*B*,*H*),(*C*,*E*)}, is it a **subgame-perfect equilibrium**?

Is this subgame a NE? NO

Subgame-Perfect Equilibrium

- Consider the NE {(*B*,*H*),(*C*,*E*)}, is it a **subgame-perfect equilibrium**?

NO

Subgame-Perfect Equilibrium

- Consider the NE {(*A*,*H*),(*C*,*F*)}, is it a **subgame-perfect equilibrium**?

Is this subgame a NE? NO

Subgame-Perfect Equilibrium

- Consider the NE {(*A*,*H*),(*C*,*F*)}, is it a **subgame-perfect equilibrium**?

NO

Subgame-Perfect Equilibrium

- Since G is its own subgame, **every subgame-perfect equilibrium is also a Nash equilibrium**

- Subgame-perfect equilibrium is a **stronger concept** than Nash equilibrium
  - Every SPE is a NE
  - Not every NE is a SPE

- Every perfect-information extensive-form game has **at least one subgame-perfect equilibrium**

Outline

- Perfect-information games in extensive form
- Strategies and equilibria
- Subgame-perfect equilibrium
- **Backward induction**
- Example

Backward Induction

- **Backward induction** is a procedure to compute the **subgame-perfect equilibrium**
- General idea:
  - Identify the equilibria in the “bottom-most” subgame trees
  - Consider that these equilibria will be played
  - Back up and consider increasingly larger trees

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

What is the NE?

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

What is the NE?

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

What is the NE?

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

What is the NE?

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Hence, $\{(A,G),(C,F)\}$, is a **subgame-perfect equilibrium**

Backward Induction

```
function BACKWARDINDUCTION (node h) returns u(h)
if h ∈ Z then
    return u(h)                                          // h is a terminal node
best_util ← −∞
forall a ∈ χ(h) do
    util_at_child ←BACKWARDINDUCTION(σ(h, a))
    if util_at_child_ρ(h) > best_util_ρ(h) then
        best_util ← util_at_child
return best_util
```

This procedure is a single depth-first traversal of the game tree. Thus, it requires time linear in the size of the game representation.

Backward Induction

function BACKWARDINDUCTION (node h) **returns** $u(h)$
if $h \in Z$ **then**
  return $u(h)$ // h is a terminal node
$best_util \leftarrow -\infty$
forall $a \in \chi(h)$ **do**
  $util_at_child \leftarrow$ BACKWARDINDUCTION$(\sigma(h, a))$
  if $util_at_child_{\rho(h)} > best_util_{\rho(h)}$ **then**
    $best_util \leftarrow util_at_child$
return $best_util$

Returns the payoff when it reaches a terminal node (end of the recursion)

Keeps track of the best payoff of a node

Iterates over all actions of a node

Recursively calls the function for a child node

Recall that each non-terminal node has an associated agent. If the agent's payoff in the child node is greater than agent's payoff in *best_util* then update *best_util*

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- Consider the extensive-form game below. **Let us use backward induction.**

Backward Induction

- For zero-sum games, Backward Induction has another name:

the minimax algorithm

- In this case, it is enough to store one number per node

- It is possible to speed up things with the “alpha-beta pruning”

Exercise

- **Escalation Game**:
  - Two countries are in the brink of war
  - Country 1 may accept the status quo or issue a threat
    - If Country 1 accepts the status quo, the game ends
  - If Country 1 threatens, Country 2 decides to either to concede or escalate the conflict
    - If Country 2 concedes, the game ends

Exercise

- **Escalation Game**:
  - If Country 2 escalates, Country 1 chooses whether to launch war or back down
    - Either way, the game ends

Exercise

- **Escalation Game**:

Exercise

- Use backward induction to find the subgame-perfect equilibrium

Exercise

Convert the game to an extensive form. Check the Nash equilibria and SPE

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, <u>0</u>
	Confess	0, -9	-6, -6

Exercise

Convert the game to an extensive form. Check the Nash equilibria and SPE

		Agent 2	
		Heads	***Tails***
Agent 1	***Heads***	-1, 1	1, -1
	Tails	1, -1	-1, 1

Thank You

rui.prada@tecnico.ulisboa.pt

TÉCNICO LISBOA

Multiagent decision making and Games in Extensive Form (Part 2)

Outline

- **Imperfect-information games in extensive form**
- Strategies and equilibria
- Mixed and Behavioral Strategies

Imperfect-information extensive-form games

- So far, we modeled agents that can **specify the actions they will take at every choice node**
  - This implies the **agents know the node they are**
  - And **all prior choices**, including those of other agents
- We called this **perfect-information games**!

Imperfect-information extensive-form games

- However, we might not want to **make such a strong assumption** about:
  - Our **agents**
  - And our **environment**

Imperfect-information extensive-form games

- In some situation, we might want to **model our agents** that:
  - **act with partial or no knowledge of the actions** taken by other agents
  - or even **agents with limited memory** of their own past actions

Imperfect-information extensive-form games

- However, we could **not model these type of agents with perfect-information agents**

- Hence, **imperfect-information games address this limitation**

Imperfect-information extensive-form games

- For example, the **following games involve hidden actions from other agents**:

Poker

Cribbage

Imperfect-information extensive-form games

- An imperfect-information game is an extensive-form game in which **each agent's choice nodes are partitioned into information sets**

- Intuitively, **if two choice nodes are in the same information set then the agent cannot distinguish between them**

Imperfect-information extensive-form games

- **Definition (Imperfect-information game):** *An imperfect-information game (in extensive form) is a tuple* $G = (N, A, H, Z, \chi, \rho, \sigma, u, I)$*, where:*

  - $(N, A, H, Z, \chi, \rho, \sigma, u)$ *is perfect-information extensive-form game; and*

  - $I = (I_1, \dots, I_n)$*, where* $I_i = (I_{i,1}, \dots, I_{i,ki})$ *is a set of equivalence classes on (i.e., a partition of)* $\{h \in H : \rho(h) = i\}$ *with the property that* $\chi(h) = \chi(h')$ *and* $\rho(h) = \rho(h')$ *whenever there exists a j for which* $h \in I_{i,j}$ *and* $h' \in I_{i,j}$.

Imperfect-information extensive-form games

- **For the choice nodes to be truly indistinguishable**
  - We require that the **set of actions at each choice node in an information set be the same** (otherwise, the agent would be able to distinguish the nodes)
- Hence, if $I_{i,j} \in I$ is an equivalence class, **we can unambiguously use the notation $\chi(\boldsymbol{I_{i,j}})$** to denote the set of actions available to agent i at any node in information set $I_{i,j}$

Imperfect-information extensive-form games

Interpretation: Agent 1 does not know whether Agent 2 has chosen *A* or *B* when he has to decide between l or r

Outline

- Imperfect-information games in extensive form
- **Strategies and equilibria**
- Mixed and Behavioral Strategies

Strategies and Equilibria

- A **pure strategy** for an agent in an imperfect-information game is:
  - **one of the available actions in each information set** of that agent

Strategies and Equilibria

- **Definition (Pure strategies):** *Let* $G = (N, A, H, Z, \chi, \rho, \sigma, u, I)$ *be a imperfect-information extensive-form game. Then the pure strategies of agent* i *consist of the Cartesian product*

$$\Pi_{I_{i,j} \in I_i} \chi(I_{i,j}).$$

Action function

Imperfect-information extensive-form games

Pure Strategies:

S_1 = {(L,l), (L,r), (R,l), (R,r)}

S_2 = {A,B}

Strategies and Equilibria

- **So how do we compute the Nash equilibria?**
  - The definition of best response is the same as we've seen so far!
  - The Nash Equilibrium (both pure and mixed) concept remains the same for imperfect-information extensive-form games

Strategies and Equilibria

- We convert an imperfect-information game to an equivalent normal-form game and compute the Nash equilibria

	A	***B***
(L,l)	0, 0	<u>2</u>, <u>4</u>
(L,r)	<u>2</u>, <u>4</u>	0, 0
(R,l)	1, <u>1</u>	1, <u>1</u>
(R,r)	1, <u>1</u>	1, <u>1</u>

Nash equilibria in this game:
{(L,l), B} and {(L,r), A}

Does this make sense?

Strategies and Equilibria

- Let us look at this subgame and **convert it into a normal-form game** where both **agents select actions simultaneously**

Coordination game

	A	***B***
l	0, 0	<u>2</u>, <u>4</u>
r	<u>2</u>, <u>4</u>	0, 0

Pure strategy Nash equilibria:
$\{l, \text{B}\}$ and $\{r, \text{A}\}$

Mixed strategy Nash equilibrium:
$\{(\frac{1}{2}, \frac{1}{2}), (\frac{1}{2}, \frac{1}{2})\}$

Strategies and Equilibria

- Converting a normal-form game to an imperfect-information extensive-form game

	C	***D***
C	-1, -1	-9, <u>0</u>
D	<u>0</u>, -9	<u>-6</u>, <u>-6</u>

Prisoner's Dilemma

Perfect-information games were not expressive enough to capture the Prisoner's Dilemma game (and many other ones)

Strategies and Equilibria

So these mappings are possible

Imperfect-information extensive-form game → Normal-form game

Normal-form game → Imperfect-information extensive-form game

Strategies and Equilibria

What happens if we apply each mapping in turn?

Imperfect-information extensive-form game → Normal-form game

Normal-form game → Imperfect-information extensive-form game

Do we end up with the same game?

Strategies and Equilibria

- Imperfect-information game to an equivalent normal-form game

	A	***B***
(L,l)	0, 0	2, 4
(L,r)	2, 4	0, 0
(R,l)	1, 1	1, 1
(R,r)	1, 1	1, 1

Strategies and Equilibria

- Normal-form game to an equivalent imperfect-information game

	A	***B***
(L,l)	0, 0	2, 4
(L,r)	2, 4	0, 0
(R,l)	1, 1	1, 1
(R,r)	1, 1	1, 1

Not the same game!

Strategies and Equilibria

- What happens if we apply each mapping in turn?
  - We might not end up with the same game
  - However, **we do get one with the same strategy space and equilibria!**

Strategies and Equilibria

- **We do get one with the same strategy space and equilibria!**

	A	***B***
(L,l)	0, 0	$\underline{2}$, $\underline{4}$
(L,r)	$\underline{2}$, $\underline{4}$	0, 0
(R,l)	1, $\underline{1}$	1, $\underline{1}$
(R,r)	1, $\underline{1}$	1, $\underline{1}$

We end up with strategically equivalent games!

Exercise

1 - Present the pure strategies

2 - Convert this game into an equivalent normal-form game

3 - Find the Nash equilibria

Outline

- Imperfect-information games in extensive form
- Strategies and equilibria
- **Mixed and Behavioral Strategies**

Mixed and Behavioral Strategies

- **Two meaningful different kinds of randomized strategies** in imperfect-information extensive-form games (and perfect-information game too)
  - Mixed strategies
  - Behavioral strategies

Mixed and Behavioral Strategies

- **Mixed strategies**: randomize over pure strategies
- **Behavioral strategy**: randomize every time an information set is encountered

Mixed and Behavioral Strategies

- **Definition (mixed strategy):** *Let* $G = (N, A, H, Z, \chi, \rho, \sigma, u, I)$ *be a imperfect-information extensive-form game. Then a mixed strategy* s_i *is any distribution over of agent* i*'s pure strategies*

$$s_i \in \Delta(A^{I_i})$$

Mixed and Behavioral Strategies

- **Definition (behavioral strategy):** *A behavioral strategy b_i is a mapping from an agent's information sets to a distribution over the actions at that information set, which is sampled independently each time the agent arrives at the information set:*

$$b_i \in [\Delta(\chi(I))]_{I \in I_i}$$

Mixed and Behavioral Strategies

For example:

Mixed strategy:
[0.6 : (*A*,*G*), 0.4 : (*B*,*H*)]

Behavioral strategy:
([0.6 : *A*, 0.4 : *B*],
[0.6 : *G*, 0.4 : *H*])

Are the strategies equivalent? Why?

Mixed and Behavioral Strategies

Are the strategies equivalent? Why?

Mixed strategy:
[0.6 : (*A*,*G*), 0.4 : (*B*,*H*)] **NO**

Behavioral strategy:
([0.6 : *A*, 0.4 : *B*],
[0.6 : *G*, 0.4 : *H*])

Mixed strategy:
[0.6 x 0.6 : (*A*,*G*), 0.6 x 0.4 : (*A*,*H*),
0.4 x 0.6 : (*B*,*G*), 0.4 x 0.4 : (*B*,*H*)] =
[0.36 : (*A*,*G*), 0.24 : (*A*,*H*),
0.24 : (*B*,*G*), 0.16 : (*B*,*H*)] **YES**

Mixed and Behavioral Strategies

- Although mixed strategy and behavioral strategy are defined differently

- In perfect-information games, **mixed strategy and behavioral strategy can emulate each other** [Kuhn, 1953]

Mixed and Behavioral Strategies

- This **can also apply to imperfect-information games,** but they need to have **perfect recall**

- **Theorem** [Kuhn, 1953]
  - In a **game of perfect recall**, any mixed strategy of a given agent can be replaced by an equivalent behavioural strategy, and any behavioural strategy can be replaced by an equivalent mixed strategy

Mixed and Behavioral Strategies

- Intuitively, a game of **perfect recall** needs to have **agents that can have full recollection of the experience** in the game
  - The agents **know all the information sets they visited previously**
  - The agents **know all the actions they have taken**

Mixed and Behavioral Strategies

Game of imperfect recall

Mixed and Behavioral Strategies

What are the pure strategies?

1: {L, R}
2: {U, D}

Game of imperfect recall

Mixed and Behavioral Strategies

What is the mixed strategy Nash equilibrium?

Note that Agent 2 has a dominant strategy (D)

Game of imperfect recall

Mixed and Behavioral Strategies

What is the mixed strategy Nash equilibrium?

Hence, agent 1's best response is *R*

Game of imperfect recall

Mixed and Behavioral Strategies

What is the mixed strategy Nash equilibrium?

(*R*, *D*) or
(0, 1),(0, 1)

Game of imperfect recall

Mixed and Behavioral Strategies

What is an equilibrium in behavioral strategy?

First note that Agent 2 still has a dominant strategy (D)

Game of imperfect recall

Mixed and Behavioral Strategies

Agent 1 randomizes every time he is in this information set

What is an equilibrium in behavioral strategy?

If Agent 1 uses a behavioral strategy (p, $1 - p$)

What is agent 1's expected utility?

Game of imperfect recall

Mixed and Behavioral Strategies

Agent 1 randomizes every time he is in this information set

What is an equilibrium in behavioral strategy?

If Agent 1 uses a behavioral strategy (p, $1 - p$)

Agent 1's expected utility is:

$1 \times p \times p$

Game of imperfect recall

Mixed and Behavioral Strategies

Agent 1 randomizes every time he is in this information set

Game of imperfect recall

What is an equilibrium in behavioral strategy?

If Agent 1 uses a behavioral strategy (p, $1 - p$)

Agent 1's expected utility is:

$p^2 + 100 \times p \times (1 - p)$

Mixed and Behavioral Strategies

Agent 1 randomizes every time he is in this information set

Agent 2 always selects *D* with probability 1 (dominant strategy)

What is an equilibrium in behavioral strategy?

If Agent 1 uses a behavioral strategy (p, $1 - p$)

Agent 1's expected utility is:

$p^2 + 100\,p\,(1 - p) + 2 \times (1 - p) =$

$-99\,p^2 + 98\,p + 2$

Game of imperfect recall

Mixed and Behavioral Strategies

Agent 1 randomizes every time he is in this information set

Game of imperfect recall

What is an equilibrium in behavioral strategy?

What is Agent 1's maximum expected utility?

$$\frac{d}{dp}(-99\,p^2 + 98\,p + 2) = 0$$

$$-198\,p + 98 = 0$$

$$p = \frac{98}{198}$$

Mixed and Behavioral Strategies

Agent 1 randomizes every time he is in this information set

Game of imperfect recall

What is an equilibrium in behavioral strategy?

$$\left(\frac{98}{198}, \frac{100}{198}\right), (0, 1)$$

Thus, the mixed strategy equilibrium can be different from the behavioral strategy equilibrium in an imperfect recall game

Thank You

rui.prada@tecnico.ulisboa.pt

Multiagent decision making: Repeated Games

Outline

- **Introduction to Repeated games**
- Finitely-repeated games
- Infinitely-repeated games
- Folk Theorem
- Replicator dynamics

Repeated Games

- **What happens if we play the same normal-form game over and over?**

Repeated Games

- Questions we will need to answer in repeated games:
  - Can agents **observe** other agent's actions?
  - Can agents **remember** the past?
  - What is the agent's **utility** for the whole game?

Repeated Games

- Some of the questions will have **different answers** for:
  - **Finitely-repeated games**
  - **Infinitely-repeated games**
- The normal-form game that we play repeatedly is called the **stage game**

Outline

- Introduction to Repeated games
- **Finitely-repeated games**
- Infinitely-repeated games
- Folk Theorem
- Replicator dynamics

Finitely-Repeated Games

- How can we analyze a **repeated game with a finite number of repetitions**?
  - We can model this game as an **extensive-form game with imperfect information**
    - At each round, **players do not know what the other players have done, but afterward, they do**
    - **The overall payoff function is additive**: the sum of payoffs in stage games

Finitely-Repeated Games

- Let us analyze the Prisoner's Dilemma in a two-stage game

	C	***D***
C	$-1, -1$	$-4, 0$
D	$0, -4$	$-3, -3$

	C	***D***
C	$-1, -1$	$-4, 0$
D	$0, -4$	$-3, -3$

Finitely-Repeated Games

Pure Strategies:

Agent 1:

$\{(C_1, C_{1,1}), (C_1, D_{1,1}), (C_1, C_{1,2}), (C_1, D_{1,2}), (C_1, C_{1,3}), (C_1, D_{1,3}), (C_1, C_{1,4}), (C_1, D_{1,4}),$
$(D_1, C_{1,1}), (D_1, D_{1,1}), (D_1, C_{1,2}), (D_1, D_{1,2}), (D_1, C_{1,3}), (D_1, D_{1,3}), (D_1, C_{1,4}), (D_1, D_{1,4})\}$

Agent 2:

$\{(C_1, C_{2,1}), (C_1, D_{2,1}), (C_1, C_{2,2}), (C_1, D_{2,2}), (C_1, C_{2,3}), (C_1, D_{2,3}), (C_1, C_{2,4}), (C_1, D_{2,4}),$
$(D_1, C_{2,1}), (D_1, D_{2,1}), (D_1, C_{2,2}), (D_1, D_{2,2}), (D_1, C_{2,3}), (D_1, D_{2,3}), (D_1, C_{2,4}), (D_1, D_{2,4}),\}$

Finitely-Repeated Games

Is the strategy profile $(D_1,D_{1,4})$, $(D_1,D_{2,4})$ a Nash equilibrium?

Finitely-Repeated Games

Is the strategy profile $(D_1,D_{1,4})$, $(D_1,D_{2,4})$ a Nash equilibrium?

Finitely-Repeated Games

Is the strategy profile $(D_1,D_{1,4})$, $(D_1,D_{2,4})$ a Nash equilibrium?

Finitely-Repeated Games

Is the strategy profile $(D_1,D_{1,4})$, $(D_1,D_{2,4})$ a Nash equilibrium?

Finitely-Repeated Games

Is the strategy profile $(D_1,D_{1,4})$, $(D_1,D_{2,4})$ a Nash equilibrium?

YES

Finitely-Repeated Games

- **Proposition:** *If the stage game G has a unique Nash equilibrium then, for any finite T, the repeated game G(T) has a unique outcome: the Nash equilibrium of G is played in every subgame-perfect stage*

Outline

- Introduction to Repeated games
- Finitely-repeated games
- **Infinitely-repeated games**
- Folk Theorem
- Replicator dynamics

Infinitely-Repeated Games

- Let us now consider a **game is repeated an infinite number of times**

- Can we represent this as an **extensive-form game**?
  - An **infinite tree**!
  - Sum of payoffs is **infinity**!

Infinitely-Repeated Games

- **Definition**: Given an infinite sequence of payoffs $r_1, r_2, \ldots$ for agent i, the **average reward** of agent i is:

$$\lim_{k \to \infty} \sum_{j=1}^{k} \frac{r_j}{k}$$

Infinitely-Repeated Games

- **Definition**: Given an infinite sequence of payoffs $r_1, r_2, \ldots$ for agent i and discount factor β with $0 < \beta < 1$, the **future discounted reward** of agent i is:

$$\sum_{j=1}^{\infty} \beta^j r_j$$

- **Interpretation**: the agent cares more about her well-being in the near term than in the long term

Infinitely-Repeated Games

- What is a **pure strategy** in an infinitely-repeated game?
  - A **choice of action at every decision point**
    - In these games, this means **an action at every stage game** (for an infinite number of actions!)

Infinitely-Repeated Games

- More formally:
  - Histories of length t:
    - $H^t = \{h^t : h^t = (a^1, \dots, a^t) \in A^t)$
    - $a^t = (a_1^t, \dots, a_n^t)$ and $a_i^t \in A_i$
  - All finite histories: $H = \bigcup H^t$
  - A pure strategy: $s_i : H \to A_i$

Infinitely-Repeated Games

- Prisoner's Dilemma
  - $A_i = \{C, D\}$
  - A history of length $t = 3$:
    - $t = 1$: (C, C)
    - $t = 2$: (C, D)
    - $t = 3$: (D, D)
  - A strategy for period 4 would specify what an agent would do after seeing $(C, C), (C, D), (D, D)$

Infinitely-Repeated Games

- Some famous strategies (repeated Prisoner's Dilemma):
  - **Tit-for-tat**: Start out cooperating. If the opponent defected, defect in the next round. Then go back to cooperation.
    - **Agent 1: C, C, C, D, C, C,…**
    - Agent 2: C, C, D, C, C, C,…

Infinitely-Repeated Games

- Some famous strategies (repeated Prisoner's Dilemma):
  - **Trigger**: Start out cooperating. If the opponent ever defects, defect forever.
    - **Agent 1: C, C, C, D, D, D,…**
    - Agent 2: C, C, D, C, C, C,…

Infinitely-Repeated Games

- **Subgame-perfect equilibrium**
  - Profile of strategies that are a **Nash equilibrium in every subgame**
    - Hence, a Nash equilibrium following every possible history
  - Repeatedly playing a Nash equilibrium of the stage game is always a **subgame-perfect equilibrium** of the repeated game

Outline

- Introduction to Repeated games
- Finitely-repeated games
- Infinitely-repeated games
- **Folk Theorem**
- Replicator dynamics

Folk Theorem

- There are many **Folk Theorems** for infinitely-repeated games
  - **All are concerned with Nash equilibria** of an infinitely repeated game
  - This result was called the Folk Theorem because it **was widely known among game theorists** in the 1950s, even though **no one had published it**

Folk Theorem

- Consider a **finite normal form game $\boldsymbol{G}$** (stage game)
- Let $\boldsymbol{a} = (\boldsymbol{a_1}, \boldsymbol{a_2}, \dots, \boldsymbol{a_n})$ **be a Nash equilibrium** of the stage game G
- If $\boldsymbol{a'} = (\boldsymbol{a'_1}, \boldsymbol{a'_2}, \dots, \boldsymbol{a'_n})$ **is such that $\boldsymbol{u_i}(\boldsymbol{a'}) > \boldsymbol{u_i}(\boldsymbol{a})$** for all i, then:
  - there **exists a discount factor $\boldsymbol{\beta}$** $(0 < \beta < 1)$, such that $\beta_i \geq \beta$ for all i, and
  - there **exists a subgame perfect equilibrium of the infinite repetition of $\boldsymbol{G}$ that has $\boldsymbol{a'}$ played in every period on the equilibrium path**

Folk Theorem

- Outline of the Proof:
  - **Play a'** as long as **every agent is also playing**
  - If **any agent ever deviates**, then **play a forever after** (Trigger strategy)
  - Check that this is a **subgame-perfect equilibrium for high enough discount factors**

Folk Theorem

- Check that this is a **subgame-perfect equilibrium for high enough discount factors**
  - **Playing a forever** (if anyone deviated) is a **Nash equilibrium** in any subgame
  - Will **someone gain by deviating from a'** if nobody has in the past?

Folk Theorem

- Check that this is a **subgame-perfect equilibrium for high enough discount factors**
  - Will **someone gain by deviating from $\boldsymbol{a}'$** if nobody has in the past?
    - **Maximum gain from deviating** (over all agents) is

$$M = \max_{i,a''} u_i(a_i'', a'_{-i}) - u_i(a')$$

    - **Minimum per-period loss from future punishment** is

$$m = \min_{i} u_i(a') - u_i(a)$$

Folk Theorem

- Check that this is a **subgame-perfect equilibrium for high enough discount factors**
  - Will **someone gain by deviating from $\boldsymbol{a}$'** if nobody has in the past?
    - If an agent deviates, given the other agents' strategies, the **maximum possible net gain** is

$$M - m\frac{\beta_i}{1-\beta_i}$$

Folk Theorem

- Check that this is a **subgame-perfect equilibrium for high enough discount factors**
  - Will **someone gain by deviating from $\boldsymbol{a}$'** if nobody has in the past?
    - **Deviation is not beneficial** if

$$M - m\frac{\beta_i}{1-\beta_i} \leq 0$$

$$\frac{M}{m} \leq \frac{\beta_i}{1-\beta_i}$$

$$\beta_i \geq \frac{M}{M+m}, \text{ for all } i$$

Infinitely-Repeated Games

- Example: **Prisoner's Dilemma – Can we sustain cooperation?**

We want to sustain cooperation

	C	***D***
C	$3, 3$	$0, 5$
D	$5, 0$	$1, 1$

Nash equilibrium of the stage game

Infinitely-Repeated Games

- Example: **Prisoner's Dilemma – Can we sustain cooperation?**
  - Agent cooperates as long as the other agent is cooperating in the past
  - Both agents defect if anyone deviates (**Trigger strategy**)

When is this an equilibrium? Is there a value of β that could sustain this equilibrium above?

	C	***D***
C	$3, 3$	$0, 5$
D	$5, 0$	$1, 1$

Infinitely-Repeated Games

- Example: **Prisoner's Dilemma – Can we sustain cooperation?**
  - **Always cooperate**: $3 + \beta 3 + \beta^2 3 + \beta^3 3 + \cdots = \frac{3}{1-\beta}$
  - **Always defect**: $5 + \beta 1 + \beta^2 1 + \beta^3 1 + \cdots = 5 + \beta \frac{1}{1-\beta}$

	C	***D***
C	$3, 3$	$0, 5$
D	$5, 0$	$1, 1$

Infinitely-Repeated Games

- Example: **Prisoner's Dilemma – Can we sustain cooperation?**
  - **Always cooperate**: $3 + \beta 3 + \beta^2 3 + \beta^3 3 + \cdots = 3 + \beta \frac{3}{1-\beta}$
  - **Always defect**: $5 + \beta 1 + \beta^2 1 + \beta^3 1 + \cdots = 5 + \beta \frac{1}{1-\beta}$
  - **Difference**: $-2 + \beta 2 + \beta^2 2 + \beta^3 2 + \cdots = -2 + \beta \frac{2}{1-\beta}$

Infinitely-Repeated Games

- Example: **Prisoner's Dilemma – Can we sustain cooperation?**
  - **Difference**: $-2 + \beta 2 + \beta^2 2 + \beta^3 2 + \cdots = \beta \frac{2}{1-\beta} - 2$
  - If we want the "**always cooperate**" to have a higher payoff:
    - $\beta \frac{2}{1-\beta} - 2 \geq 0$
    - $\beta \geq \frac{1}{2}$
    - **Interpretation**: if we want to sustain cooperation, the agent needs to care about tomorrow at least as half as much as it cares about today!

Exercise

- Could we sustain cooperation in this infinitely-repeated game?

	C	***D***
C	$3, 3$	$0, 10$
D	$10, 0$	$1, 1$

Defection is more attractive!

Repeated Games

- Agents can condition future decisions based on past actions
- This generates many equilibria: Folk Theorems
- Check “The Evolution of Trust”, Nicky Case
  - https://ncase.me/trust/

Outline

- Introduction to Repeated games
- Finitely-repeated games
- Infinitely-repeated games
- Folk Theorem
- **Replicator dynamics**

Evolutionary Learning and other Large-population models

- Consider large populations
- Model the evolution of behavior of the whole population
- Standard model technique in biology, social networks, human population behavior.

MULTIAGENT SYSTEMS

Algorithmic, Game-Theoretic, and Logical Foundations

Yoav Shoham, Kevin Leyton-Brown, 2008

Evolutionary Learning and other Large-population models

The *replicator dynamic* models a population undergoing frequent replicator interactions.

Replicator dynamic

Basic interaction between any two agents

	A	***B***
A	x, x	u, v
B	v, u	y,y

agents have no distinct roles (symmetric game)

Evolutionary Learning and other Large-population models

Replicator dynamic

	A	***B***
A	x, x	u, v
B	v, u	y,y

- $\varphi_t(A)$ – number of agents playing A at time t
- $\theta_t(A) = \frac{\varphi_t(A)}{\sum_a \varphi_t(a)}$ - ratio of agents playing A at time t
- $u_t(a) = \sum_b \theta_t(b) u(a, b)$

Evolutionary Learning and other Large-population models

Replicator dynamic

- $u_t(a) = \sum_b \theta_t(b) u(a,b)$
- $u_t^* = \sum_a \theta_t(a) u_t(a)$

The number of agents change proportionally to their utility.

- $\dot{\varphi}_t(a) = \varphi_t(a) u_t(a)$
- $\theta_t(A) = \frac{\varphi_t(A)}{\sum_a \varphi_t(a)}$

- $\dot{\theta}_t(A) = \frac{\dot{\varphi}_t(A) \sum_a \varphi_t(a) - \varphi_t(A) \sum_a \dot{\varphi}_t(a)}{(\sum_a \varphi_t(a))^2} = \theta_t(a)[u_t(a) - u_t^*]$

Replicator dynamics

```
def dynreplicator(g,th):
    va = g.payoff_matrices[0]@th
    v = th@va
    dth = th * (va-v)

    return dth

ddx = np.zeros((100,100))
ddy = np.zeros((100,100))

for kk in Games.keys():
    g = Games[kk]
    for ix in np.linspace(0, .99, 10,):
        for iy in np.linspace(0, .99, 10):
            dth = dynreplicator(g,[ix,iy])
            ddx[int(ix*10),int(iy*10)],ddy[int(ix*10),int(iy*10)] = dth
    plt.figure()
    plt.subplot(1,2,1)
    plt.quiver(xx,yy,ddx,ddy)
    plt.title(kk)

    th = [0.7,0.3]
    TH = []
    for ii in np.arange(0,20,dt):

        dth = dynreplicator(g,th)
        th = np.clip(th + dth * dt, 0, 1)
        th /= np.sum(th)
        TH += [th]
    plt.subplot(1,2,2)
    plt.plot(TH)
    plt.title(kk)
```

Replicator dynamics

Definition 7.7.2 (Steady state) *A* steady state *of a population using the replicator dynamic is a population state* θ *such that for all* $a \in A$, $\dot{\theta}(a) = 0$.

Definition 7.7.3 (Stable steady state) *A steady state* θ *of a replicator dynamic is* stable *if for every neighborhood* U *of* θ *there is another neighborhood* U' *of* θ *such* state *that if* $\theta_0 \in U'$ *then* $\theta_t \in U$ *for all* $t > 0$.

Definition 7.7.4 (Asymptotically stable state) *A steady state* θ *of a replicator dynamic is* asymptotically stable *if it is stable, and in addition if for every neighborhood* U *of* θ *it is the case that if* $\theta_0 \in U$ *then* $\lim_{t\to\infty} \theta_t = \theta$.

What is the equilibria? Is it stable?

	A	B
A	$0, 0$	$1, 1$
B	$1, 1$	$0, 0$

Figure 7.9: The Anti-Coordination game.

Anti-Coordination Game

- $u(a)=0\,\theta + 1\,(1-\theta) = 1-\theta$
- $u(b)=1\,\theta + 0\,(1-\theta) = \theta$

- $u^*=(1-\theta)\,\theta + \theta\,(1-\theta) = 2\,(1-\theta)\,\theta$

- $d\theta = \theta(1-\theta-2\,\theta(1-\theta)) = \theta(1-3\,\theta+2\,\theta^2)$

- $d\theta=0$ for $\theta = 0.5$
- $d\theta>0$ for $\theta<0.5$
- $d\theta>0$ for $\theta>0.5$

Stag-Hare

	Stag	Hare
Stag	10, 10	1, 8
Hare	8, 1	5, 5

- What are the Nash Eq?
- Is there an assymptotic equilibria?

Stag-Hare

	Stag	Hare
Stag	10, 10	1, 8
Hare	8, 1	5, 5

- $u(s)=10\,\theta+1\,(1-\theta) = 9\,\theta+1$
- $u(h)=8\,\theta+5(1-\theta) = 3\,\theta+5$
- $u^* = (9\,\theta+1)\,\theta+(3\,\theta+5)(1-\theta)=6\,\theta^2-\theta+5$
- $u(s)-u^*=(9\,\theta+1)-(6\,\theta^2-\theta+5)=-6\theta^2+10\theta-4$
- $u(s)-u^*=0 \Rightarrow \theta=1 \;\vee\; \theta = 2/3$

Stag-Hare

- u(s)=10 θ+ 1 (1- θ) = 9 θ+1
- u(h)=8 θ+5(1- θ) = 3 θ+3
- u* = (9 θ+1) θ+(3 θ+5)(1- θ)=6 θ^2- θ+5
- u(s)-u*=(9 θ+1)-(6 θ^2- θ+5)=-6θ^2+10θ-4
- u(s)-u*=0 => θ=1 V θ = 2/3

Replicator dynamics

- In a given game the populations will converge to the Nash equilibria?
- What if there is more than one?
- And if there is none?

Replicator dynamics

Theorem 7.7.5 *Given a normal-form game* $G = (\{1, 2\}, A = \{a_1, \ldots, a_k\}, u)$, *if the strategy profile* (S, S) *is a (symmetric) mixed strategy Nash equilibrium of* G *then the population share vector* $\theta = (S(a_1), \ldots, S(a_k))$ *is a steady state of the replicator dynamic of* G.

Theorem 7.7.6 *Given a normal-form game* $G = (\{1, 2\}, A\{a_1, \ldots, a_k\}, u)$ *and a mixed strategy* S, *if the population share vector* $\theta = (S(a_1), \ldots, S(a_k))$ *is a stable steady state of the replicator dynamic of* G, *then the strategy profile* (S, S) *is a mixed strategy Nash equilibrium of* G.

Replicator dynamics

Definition 7.7.7 (Trembling-hand perfect equilibrium) *A mixed strategy* S *is a* (trembling-hand) perfect equilibrium *of a normal-form game* G *if there exists a sequence* $S_0, S_1, \ldots$ *of fully mixed-strategy profiles such that* $\lim_{n\to\infty} S_n = S$, *and such that for each* S_k *in the sequence and each player* i, *the strategy* s_i *is a best response to the strategies* s_{k-i}.

Theorem 7.7.8 *Given a normal-form game* $G = (\{1, 2\}, A, u)$ *and a mixed strategy* S, *if the population share vector* $\theta = (S(a_1), \ldots, S(a_k))$ *is an asymptotically stable steady state of the replicator dynamic of* G, *then the strategy profile* (S, S) *is a Nash equilibrium of* G *that is trembling-hand perfect and isolated.*

Evolutionarily stable strategies

Unlike the steady states discussed earlier, it does not require the replicator dynamic; rather it is a static solution concept.
Roughly speaking, an evolutionarily stable strategy is a mixed strategy that is "resistant to invasion" by new strategies.

Definition 7.7.10 (Weak ESS) S *is a* weak evolutionarily weak stable strategy *if and only if for some* $\varrho > 0$ *and for all* S' *it is the case that either* $u(S, S) > u(S', S)$ *holds, or else both* $u(S, S) = u(S', S)$ *and* $u(S, S') \geq u(S', S')$ *hold.*

Evolutionarily stable strategies

Unlike the steady states discussed earlier, it does not require the replicator dynamic; rather it is a static solution concept.
Roughly speaking, an evolutionarily stable strategy is a mixed strategy that is "resistant to invasion" by new strategies.

Definition 7.7.10 (Weak ESS) S *is a* weak evolutionarily weak stable strategy *if and only if for some* $\varrho > 0$ *and for all* S' *it is the case that either* $u(S, S) > u(S', S)$ *holds, or else both* $u(S, S) = u(S', S)$ *and* $u(S, S') \geq u(S', S')$ *hold.*

Theorem 7.7.11 *Given a symmetric two-player normal-form game* $G = (\{1, 2\}, A, u)$ *and a mixed strategy* S*, if* S *is an evolutionarily stable strategy then* (S, S) *is a Nash equilibrium of* G.

Theorem 7.7.12 *Given a symmetric two-player normal-form game* $G = (\{1, 2\}, A, u)$ *and a mixed strategy* S*, if* (S, S) *is a strict (symmetric) Nash equilibrium of* G*, then* S *is an evolutionarily stable strategy.*

Theorem 7.7.13 *Given a symmetric two-player normal-form game* $G = (\{1, 2\}, A, u)$ *and a mixed strategy* S*, if* S *is an evolutionarily stable strategy then it is an asymptotically stable steady state of the replicator dynamic of* G.

Matching Pennies

Prisioner dilema

Hawk-Dove

Hawk-Dove
1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
Hawk-Dove
1.0
0.8
0.6
0.4
0.2
0.0
0
500
1000
1500
2000

Thank You

rui.prada@tecnico.ulisboa.pt
manuel.lopes@tecnico.ulisboa.pt
alberto.sardinha

Markov decision process (MDP)

AASMA 2024

Alberto Sardinha jose.alberto.sardinha@tecnico.ulisboa.pt
Rui Prada rui.prada@tecnico.ulisboa.pt
Manuel Lopes manuel.lopes@tecnico.ulisboa.pt

Outline

- **Single-agent learning**

Markov decision process (MDP)

- A framework for **sequential decision making** of **a single agent**

- **Markovian transition model** and **fully observable**
  - i.e., we assume it verifies the Markov property

- **Planning horizon** can be infinite

Markov decision process (MDP)

Markov decision process (MDP)

- We can formally define an MDP with following elements:
  - **Discrete time** $t = 0, 1, 2, \ldots$
  - A **discrete set of states** $s \in S$
  - A **discrete set of actions** $a \in A$
  - A **stochastic transition model** $P(s'|s, a)$
    - the world transitions stochastically to state s' when the agent takes action a at state s
  - A **reward function** $R: S \times A \rightarrow \mathbb{R}$
    - An agent receives a reward $R(s, a)$ when it takes action a at state s

Markov decision process (MDP)

- **Definition**: the **state-value function** of a state s under a policy π is the expected return the agent can receive when starting in state s and then following policy π :

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

Definition: the **action-value function (Q-values)** of taking an action a in state s under a policy π is the expected return the agent can receive when starting in state s, taking action a, and then following policy π :

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

Markov decision process (MDP)

- A **policy π is defined to be better than or equal to a policy π'** if the expected return of π is greater or equal to expected return of π':

$$\pi \geq \pi' \text{ if and only if } V^{\pi}(s) \geq V^{\pi'}(s), \text{ for all } s \in S$$

- **There is always at least one policy** that is better than or equal to all other policies, which is the **optimal policy**
  - Note that an MDP might have more than one optimal policy
  - We denote all the optimal polices by π^*

Markov decision process (MDP)

- These policies π^* share the same state-value function, called **optimal state-value function**, with the following definition:

$$V^*(s) = \max_{\pi} V^{\pi}(s), \text{ for all } s \in S$$

- These policies π^* also share the same action-value function, called **optimal action-value function**, with the following definition:

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a), \text{ for all } s \in S \text{ and } a \in A$$

Value Iteration

- We initialize arbitrarily a state-value function (e.g., with zeros, ones, etc.)

- Then we iteratively apply the Bellman equation turned into an assignment operation:

$$Q(s,a) \coloneqq R(s,a) + \gamma \sum_{s' \epsilon S} P(s'|s,a)V(s'),\ \forall s, \forall a$$

$$V(s) \coloneqq \max_{a \in A} Q(s,a), \forall s$$

- Repeat the above two equations until V does not change significantly between two consecutive steps

Value Iteration

- Value iteration converges to the optimal Q^* for any initialization
- After computing the optimal Q^*, we can extract the policy as follows:

$$\pi^*(s) \in \operatorname*{argmax}_{a \in A} Q^*(s, a)$$

Markov decision process (MDP)

- What happens if we **do not know the stochastic transition model** and **reward function**?

  - We can use **Reinforcement Learning**

Reinforcement Learning

- Why not **interact with the environment**?

Reinforcement Learning

- Why not **interact with the environment**?
  - At each time step t
    - The agent observes the state s_t
    - The agent takes action a_t
    - The agent observes a reward R_t and the new state s_{t+1}

Reinforcement Learning

- Thus, my data point at each iteration is:

$$(s_t, a_t, R_t, s_{t+1})$$

- And what is the agent's goal?
  - Compute the **optimal policy of the MDP** with the data points

- Today, we focus on the most famous RL algorithm: **Q-learning**

Reinforcement Learning

- We start with some estimate Q
- Initialize current state s
- Loop for each step:
  - choose some action a (e.g., using ϵ-greedy)
  - Take action a and observe next state s' and reward r
  - Update Q estimate according to

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

  - $s \leftarrow s'$

Q-learning

Reinforcement Learning

- Important observations about Q-Learning:
  - Update depends on previous estimate (*bootstrap*)
  - Update rule **propagates** reward information
  - To compute Q, algorithm must visit **every** state-action

Example

- We have the following MDP:
  - $S = \{1, 2, 3\}$
  - $A = \{left, right\}$
  - $\boldsymbol{P(s'|s, a = left)} = ?$
  - $\boldsymbol{P(s'|s, a = right)} = ?$
  - $\boldsymbol{R(s, a)} = ?$
  - $\gamma = 0.9, \alpha = 0.3$

Example

- We start with some estimate Q

$$Q = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

- Initialize current state s

$$s \rightarrow 1$$

Example

- First iteration (current state $s \rightarrow 1$)
  - choose some action a
    - $a \rightarrow left$
  - Take action a and observe next state s' and reward r
    - $s' \rightarrow 1$
    - $r \rightarrow 0$
  - Update Q estimate according to
    - $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
    - $Q(1, left) \leftarrow 1 + 0.3[0 + 0.9 \times 1 - 1]$
    - $Q(1, left) \leftarrow 0.97$

Example

- Updated Q

$$Q = \begin{bmatrix} 0.97 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Example

- Second iteration (current state $s \rightarrow 1$)
  - choose some action a
    - $a \rightarrow right$
  - Take action a and observe next state s' and reward r
    - $s' \rightarrow 2$
    - $r \rightarrow 0$
  - Update Q estimate according to
    - $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
    - $Q(1, right) \leftarrow 1 + 0.3[0 + 0.9 \times 1 - 1]$
    - $Q(1, right) \leftarrow 0.97$

Example

- Updated Q

$$Q = \begin{bmatrix} 0.97 & 0.97 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Example

- Third iteration (current state $s \rightarrow 2$)
  - choose some action a
    - $a \rightarrow left$
  - Take action a and observe next state s' and reward r
    - $s' \rightarrow 1$
    - $r \rightarrow 0$
  - Update Q estimate according to
    - $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
    - $Q(2, left) \leftarrow 1 + 0.3[0 + 0.9 \times 0.97 - 1]$
    - $Q(2, left) \leftarrow 0.9619$

Example

- Updated Q

$$Q = \begin{bmatrix} 0.97 & 0.97 \\ 0.9619 & 1 \\ 1 & 1 \end{bmatrix}$$

Example

- Updated Q (after many iterations)

$$Q = \begin{bmatrix} 6.86 & 7.99 \\ 7.22 & 8.91 \\ 9.2 & 10 \end{bmatrix}$$

- This is Q^* with Value Iteration

$$Q^* = \begin{bmatrix} 6.89 & 7.66 \\ 7.08 & 8.73 \\ 9.07 & 9.95 \end{bmatrix}$$

Example

- After computing Q, we can extract the policy as follows:

$$\pi^*(s) \in \underset{a \in A}{\operatorname{argmax}}\, Q(s,a)$$

$$\pi^* = \begin{bmatrix} right \\ right \\ right \end{bmatrix}$$

$$\pi^* = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

Example

```python
import numpy as np
np.set_printoptions(precision=2, suppress=True)

# States
S = ['1', '2', '3']

# Actions
A = ['L', 'R']

# Transition probabilities

L = np.array([[1.0, 0.0, 0.0],
              [0.8, 0.2, 0.0],
              [0.0, 0.8, 0.2]])

R = np.array([[0.2, 0.8, 0.0],
              [0.0, 0.2, 0.8],
              [0.0, 0.0, 1.0]])

P = [L, R]

# Reward function

R = np.array([[0.0, 0.0],
              [0.0, 0.0],
              [1.0, 1.0]])

gamma = 0.9
```

Example

```
def egreedy(Q,state,eps):

    p = np.random.random()

    if p < eps:
        action = np.random.choice(num_actions)
    else:
        action = np.argmax(Q[state,:])

    return action
```

Example

```
STEPS = 1000000
num_actions = len(A)
num_states = len(S)
ALPHA = 0.3

# Initialize Q-values
Q = np.ones((num_states, num_actions))

# Initialize current state
state = 0

for t in range(STEPS):

    # choose action
    action = egreedy(Q,state,0.05)

    # choose next state
    next_state = np.random.choice(num_states, p=P[action][state, :])

    # obtain reward
    reward = R[state,action]

    # Update Q
    Q[state, action] = Q[state, action] + ALPHA*(reward + gamma*max(Q[next_state, :]) - Q[state, action])

    state = next_state

print(Q)
```

TÉCNICO LISBOA

Multiagent Learning

Reference: www.marl-book.com
Chapter 6 – 6.1, 6.2, 6.3

AASMA 2024

Manuel Lopes
manuel.lopes@tecnico.ulisboa.pt

Multiagent Learning Applications

Applications [edit]

Multi-agent reinforcement learning has been applied to a variety of use cases in science and industry:

- Broadband cellular networks such as 5G[32]
- Content caching[32]
- Packet routing[32]
- Computer vision[33]
- Network security[32]
- Transmit power control[32]
- Computation offloading[32]
- Language evolution research[34]
- Global health[35]
- Integrated circuit design[36]
- Internet of Things[32]
- Microgrid energy management[37]
- Multi-camera control[38]
- Autonomous vehicles[39]
- Sports analytics[40]
- Traffic control[41] (Ramp metering[42])
- Unmanned aerial vehicles[43][32]
- Wildlife conservation[44]

https://en.wikipedia.org/wiki/Multi-agent_reinforcement_learning

Learning to Play: The Multi-Agent Reinforcement Learning in MalmÖ (MARLÖ) Competition

Learning to Play: The Multi-Agent Reinforcement Learning in MalmÖ (MARLÖ) Competition

- Define the states
- Define the actions
- Define the reward
- Define the transitions

https://irobotics.aalto.fi/category/deep-multi-agent-reinforcement-learning-for-decision-making-in-autonomous-driving-systems/

- Define the states
- Define the actions
- Define the reward
- Define the transitions

https://christinemcleavey.com/multi-agent-training-lessons-learned-from-training-3-separate-competing-networks/

- Define the states
- Define the actions
- Define the reward
- Define the transitions

AlphaStar

https://deepmind.google/discover/blog/alphastar-grandmaster-level-in-starcraft-ii-using-multi-agent-reinforcement-learning/

AlphaStar

- Define the states
- Define the actions
- Define the reward
- Define the transitions

https://deepmind.google/discover/blog/alphastar-grandmaster-level-in-starcraft-ii-using-multi-agent-reinforcement-learning/

https://www.deepstack.ai/

Outline

- **Stochastic games**
- Independent learning
- Learning in repeated games
- Multiagent learning
  - Joint-Action Learners
  - Agent-Modelling

Stochastic Games

- In repeated games, agents played the same stage game over and over again

Prisoner's Dilemma

	C	***D***
C	$3,3$	$0,5$
D	$5,0$	$1,1$

	C	***D***
C	$3,3$	$0,5$
D	$5,0$	$1,1$

	C	***D***
C	$3,3$	$0,5$
D	$5,0$	$1,1$

Stochastic Games

- What if agents **repeatedly played several normal-form games from a collection**

Stag Hunt

	A	*B*
X	$3, 3$	$0, 2$
Y	$2, 0$	$1, 1$

Prisoner's Dilemma

	A	*B*
X	$3, 3$	$0, 5$
Y	$5, 0$	$1, 1$

Battle of Sexes

	A	*B*
X	$2, 1$	$0, 0$
Y	$0, 0$	$1, 2$

And **the particular game played** at any given iteration **depends probabilistically on the previous game played and on the actions taken by all agents** in that game.

Stochastic Games

- **Definition (Stochastic game)**: *A stochastic game (also known as a Markov game) is a tuple* (Q, N, A, P, r)*, where:*
  - Q *is a* ***finite set of games (states)****;*
  - N *is a* ***finite set of*** $\boldsymbol{n}$ ***agents****;*
  - $A = A_1 \times \cdots \times A_n$*, where* A_i *is a* ***finite set of actions*** *available to agent* i*;*
  - $P: Q \times A \times Q \longmapsto [0,1]$ *is* ***the transition probability function****;* $P(q, a, \hat{q})$ *is the probability of transitioning from state* q *to state* $\hat{q}$ *after action profile* a*; and*
  - $R = r_1, \ldots, r_n$*, where* $r_i: Q \times A \longmapsto \mathbb{R}$ *is a* ***real-valued payoff function*** *for agent* i*.*

Stochastic Games

- A **stochastic game is generalization** of a:
  - **Repeated game**, where Q has **only one game** (one state)
  - **Markov decision process**, where N has **only one agent**

Learning in Markov Games

- Independent – cannot see others, other agents are considered part of the environment
- Centralized – even if there are several entities a central entity decides everything
- Joint-Action Learners – agents are aware of others, they will learn and decide what to do based on a given criteria, e.g. equilibria, worst-case, …
- Agent Modelling –agents are aware of others, they will learn and decide what to do based on a model of the others agents

Outline

- Stochastic games
- **Independent learning**
- Learning in repeated games
- Multiagent learning
  - Joint-Action Learners
  - Agent-Modelling

Independent Learning

- In a Stochastic game, **each agent can learn separately**
  - Ignoring the presence of the other agents in the system
- More specifically, each agent can **treat the other agents as part of its environment**
  - Thus, not trying to model the other agents or predict their actions

Independent Learning

- For example, agent 1 and agent 2 can use Q-learning to learn their policies.
  - The state definition of agent 1 could include: position of agent 1, position of agent 2, position of the ball, etc.

Independent Learning

- However this approach is **inherently flawed**:
  - In an environment with multiple learning agents, the (hypothetical) **transition model** $p(s'|s, a_i)$ of agent i **may be changing continuously** due to the policy of the other agents (who are also learning)
  - the **convergence of Q-learning** relies on an underlying **transition model** that is **stationary**
    - stationary = does not change with time

Independent Learning

- Although independent **Q-learning cannot be justified theoretically**, the method has been **employed in practice with reported success**
  - Mataric. M. J. (1994). Reward Functions for Accelerated Learning. In *Proc. 11th Int. Conf. on Machine Learning*, San Francisco, CA.
  - Sen, S., Sekaran, M., and Hale, J. (1994). Learning to coordinate without sharing information. In *Proc. 12th Nation. Conf. on Artificial Intelligence*, Seattle, WA.
  - Tan, M. (1993). Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proc. 10th Int. Conf. on Machine Learning*, Amherst, MA.

Independent Learning

- **Claus, C. and Boutilier, C. (1998).** The dynamics of reinforcement learning in cooperative multiagent systems. In *Proc. 15th Nation. Conf. on Artificial Intelligence*, Madison, WI.
  - The authors examine **the conditions under which independent Q-learning leads to individual policies that form a Nash equilibrium** in a **single state coordination problem**
  - However, the **resulting equilibrium may not be Pareto optimal**

Independent Learners

- Initialize Q^i for each agent i
- Initialize current state s^i for each agent i
- Loop for each step:
  - Loop for each agent:
    - choose some action a^i (e.g., using ϵ-greedy)
  - Take joint action $a^1, a^{\cdots}, a^n$ and observe next state s'^i and reward r^i
  - Update Q estimate according to
    $$Q(s^i, a^i) \leftarrow Q(s^i, a^i) + \alpha[r + \gamma \max_b Q(s'^i, b) - Q(s^i, a^i)]$$
  - $s^i \leftarrow s'^i$

Examples

- Repeated game of Prisioner's Dilemma

		Prisoner 2	
		Colaborate	***Defect***
Prisoner 1	***Colaborate***	-1, -1	-9, 0
	Defect	0, -9	-6, -6

- Repeated game of Matching Pennies

		Agent 2	
		Heads	***Tails***
Agent 1	***Head***	1, -1	-1, 1
	Tails	-1, 1	1, -1

Independent Learning Examples

- Q1=[0,0]
- Q2=[0,0]

- Play C, C

- Q1[C]=0+0.1*(-1-0)=-0.1
- Q2[C]=0+0.1*(-1-0)=-0.1

		Prisoner 2	
		Colaborate	***Defect***
Prisoner 1	***Colaborate***	-1, -1	-9, 0
	Defect	0, -9	-6, -6

Independent Learning Examples

- Q1=[-0.1,0]
- Q2=[-0.1,0]

- Play C, D

- Q1[C]=-0.1+0.1*(-9-(-0.1))=-0.99
- Q2[D]=0+0.1*(0-0)=0

		Prisoner 2	
		Colaborate	***Defect***
Prisoner 1	***Colaborate***	-1, -1	-9, 0
	Defect	0, -9	-6, -6

Independent Learning Examples

- Q1=[-0.99,0]
- Q2=[-0.1,0]

- Play D, D

- Q1[D]=-0.99+0.1*(-6-(-0.99))=-1,491
- Q2[D]=0+0.1*(-6-0)=-0.6

		Prisoner 2	
		Cooperate	***Defect***
Prisoner 1	***Cooperate***	-1, -1	-9, 0
	Defect	0, -9	-6, -6

Independent Learning Examples

- Q1=[-0.99, -1,491]
- Q2=[-0.1,-0.6]

- Play D, D

- Q1[D]=-0.99+0.1*(-6-(-0.99))=-1,491
- Q2[D]=0+0.1*(-6-0)=-0.6

		Prisoner 2	
		Colaborate	***Defect***
Prisoner 1	***Colaborate***	-1, -1	-9, 0
	Defect	0, -9	-6, -6

Independent Learning Examples

Independent Learning Examples

- Q1=[0,0]
- Q2=[0,0]

- Play H, H

- Q1[H]=0+0.1*(1-0)=-.1
- Q2[H]=0+0.1*(-1-0)=-0.1

		Agent 2	
		Heads	***Tails***
Agent 1	***Head***	1, -1	-1, 1
	Tails	-1, 1	1, -1

Independent Learning Examples

- Q1=[.1,0]
- Q2=[-.1,0]

- Play H, T

- Q1[H]=0.1+0.1*(-1-0.1)=-0.01
- Q2[T]=0+0.1*(1-0)=0.1

		Agent 2	
		Heads	***Tails***
Agent 1	***Head***	1, -1	-1, 1
	Tails	-1, 1	1, -1

Independent Learning Examples

- Q1=[-0.01,0]
- Q2=[-.1,0.1]

- Play H, T

- Q1[H]=
- Q2[T]=

		Agent 2	
		Heads	***Tails***
Agent 1	***Head***	1, -1	-1, 1
	Tails	-1, 1	1, -1

Independent Learning Examples

- Q1=[-0.01,0]
- Q2=[-.1,0.1]

- Play H, T

- Q1[H]=
- Q2[T]=

Agent 2

		Heads	***Tails***
Agent 1	***Head***	1, -1	-1, 1
	Tails	-1, 1	1, -1

Independent Learning in Markov Games

Imagine two bears that can get some honey from the bees or hunt a large animal. The honey they can get alone (+2 if alone, +1 if they go together). The large animal needs the collaboration of the two (they get 0 if going alone, +5 if they go together).

Define a Markov Game for this problem. Consider only two actions ←, C - capture, and →. They hunt/gather if they are in the target location and do the capture action.

The game restarts when one bear tries to capture one prey.

Independent Learning in Markov Games

We are using the symmetry and ignore the repeated states, for instance, (1,2) and (2,1)

Independent Learning in Markov Games

What will then learn if they always move to the left or always to the right?
Does it depend on what the others are doing?

Independent Learning in Markov Games

What will then learn if they always move to the left or always to the right?

If they both go left they will get the honey receiving +1
If they both go right they will capture the large prey both receiving +5
If one goes right and the other left, the one moving right gets 0 the one moving left gets +2

Independent Learning in Markov Games

Does it depend on what the others are doing?

Yes, the values they receive include the impact of the actions of others. Let's verify and compute the Q values.

Independent Learning in Markov Games

If white bear goes to the right then if the black bear always goes left it gets:

$Q(1,\leftarrow) = 0 + \gamma \max_b\ Q(0,b)$

$Q(0,C)=2$

$Q(1,\leftarrow) = 2\gamma$

If both go to the left the black bear gets:

$Q(1,\leftarrow) = 0 + \gamma \max_b\ Q(0,b)$

$Q(0,C)=1$

$Q(1,\leftarrow) = \gamma$

Independent Learning in Markov Games

Imagine Black bear is in cell 2 and white bear is in cell 1. What should Black bear do?

Wait
Capture
Go to the honey?

Independent Learning in Markov Games

Imagine Black bear is in cell 2 and white bear is in cell 1. What should Black bear do?

Wait
Capture
Go to the honey?

Black bear has no information about the location of White bear so unless they have practiced a lot this policy there is not enough information to decide.

→a better system would not be independent learning but distributed learning with full observation, i.e. the agents know the state of the other agents.

Independent Learning in Markov Games

With full observation

- Q((1,2),C) = ?
- Q((2,2),C) = ?

Independent learning

Q(1,C)=?

Q(2,C)=?

Learning in Markov Games

- a better system would not be independent learning but distributed learning with full observation, i.e. the agents know the state of the other agents.

- And if they see the actions of others?

Outline

- Stochastic games
- Learning in repeated games
- Independent learning
- **Multiagent learning**
  - Joint-Action Learners
  - Agent-Modelling

Learning in Multi-Agent Systems

What should the criteria be?

- Assume the others are doing their best
- Are they considering my preferences
- Find a stable equilibria
- Improve the worst case scenario
- Ignore others
- Try to predict others

Goalkeeper

- Player 1 (on the left can move to the cells on the right with actions up, middle, bottom). Winning if Player 2 is not on the cell
- Player 2 (on the right, wants to block Player 1, moving up, down, or center)

Independent Learning

Agent 1
Q(1, ↑)
Q(1,→)
Q(1, ↓)

Agent 2
Q(2, ↑)
Q(2, →)
Q(2, ↓)
Q(3, ↑)
Q(3, →)
Q(3, ↓)
Q(4, ↑)
Q(4, →)
Q(4, ↓)

Agents do not observe the others

Agent 1	Agent 2
Q((1,2), ↑)	Q((1,2), ↑)
Q((1,2),→)	Q((1,2), →)
Q((1,2), ↓)	Q((1,2), ↓)
Q((1,3), ↑)	Q((1,3), ↑)
Q((1,3),→)	Q((1,3), →)
Q((1,3), ↓)	Q((1,3), ↓)
Q((1,4), ↑)	Q((1,4), ↑)
Q((1,4),→)	Q((1,4), →)
Q((1,4), ↓)	Q((1,4), ↓)

Agents observe the others as part of the environment, without assuming any intentionally by the other

Joint Action-Learning

Agent 1/Agent 2	Agent 1/Agent 2	Agent 1/Agent 2
$Q((1,2), \uparrow \uparrow)$	$Q((1,3), \uparrow \uparrow)$	$Q((1,4), \uparrow \uparrow)$
$Q((1,2), \rightarrow \uparrow)$	$Q((1,3), \rightarrow \uparrow)$	$Q((1,4), \rightarrow \uparrow)$
$Q((1,2), \downarrow \uparrow)$	$Q((1,3), \downarrow \uparrow)$	$Q((1,4), \downarrow \uparrow)$
$Q((1,2), \uparrow \rightarrow)$	$Q((1,3), \uparrow \rightarrow)$	$Q((1,4), \uparrow \rightarrow)$
$Q((1,2), \rightarrow \rightarrow)$	$Q((1,3), \rightarrow \rightarrow)$	$Q((1,4), \rightarrow \rightarrow)$
$Q((1,2), \downarrow \rightarrow)$	$Q((1,3), \downarrow \rightarrow)$	$Q((1,4), \downarrow \rightarrow)$
$Q((1,2), \uparrow \downarrow)$	$Q((1,3), \uparrow \downarrow)$	$Q((1,4), \uparrow \downarrow)$
$Q((1,2), \rightarrow \downarrow)$	$Q((1,3), \rightarrow \downarrow)$	$Q((1,4), \rightarrow \downarrow)$
$Q((1,2), \downarrow \downarrow)$	$Q((1,3), \downarrow \downarrow)$	$Q((1,4), \downarrow \downarrow)$

Both agents have Q functions with the same shape, but each one computes them individually

Joint Action-Learning

Agent 1	Agent 2
$Q((1,2), \uparrow \uparrow) = -1$	$Q((1,2), \uparrow \uparrow) = 1$
$Q((1,2), \rightarrow \uparrow) = 1$	$Q((1,2), \rightarrow \uparrow) = -1$
$Q((1,2), \downarrow \uparrow) = 1$	$Q((1,2), \downarrow \uparrow) = -1$
$Q((1,2), \uparrow \rightarrow)=1$	$Q((1,2), \uparrow \rightarrow)=-1$
$Q((1,2), \rightarrow \rightarrow)=-1$	$Q((1,2), \rightarrow \rightarrow)=1$
$Q((1,2), \downarrow \rightarrow)=1$	$Q((1,2), \downarrow \rightarrow)=-1$
$Q((1,2), \uparrow \downarrow)=1$	$Q((1,2), \uparrow \downarrow)=-1$
$Q((1,2), \rightarrow \downarrow)=1$	$Q((1,2), \rightarrow \downarrow)=-1$
$Q((1,2), \downarrow \downarrow)=-1$	$Q((1,2), \downarrow \downarrow)=1$

Both agents have Q functions with the same shape, but each one computes them individually

Learning in Markov Games

Joint Action Learners

- Initialize Q^i for each agent i

$$Q(s, a^1, a^{\cdots}, a^n)$$

- Initialize current state s
- Loop for each step:
  - Loop for each agent:
    - choose some action a^i (can we use ϵ-greedy over the Q values?)
  - Take joint action $a^1, a^{\cdots}, a^n$ and observe next state s'^i and reward r^i
  - Update Q estimate according to

$$Q(s^i, a^i) \leftarrow Q(s^i, a^i) + \alpha[r + \gamma \max_b Q(s'^i, ?) - Q(s^i, a^i)]$$

  - $s^i \leftarrow s'^i$

Joint Action Learners

- Initialize Q^i for each agent i

$$Q(s, a^1, a^{\cdots}, a^n)$$

- Initialize current state s
- Loop for each step:
  - Loop for each agent:
    - choose some action a^i (can we use ϵ-greedy over the Q values?)
  - Take joint action $a^1, a^{\cdots}, a^n$ and observe next state s'^i and reward r^i
  - Update Q estimate according to

$$Q(s^i, a^i) \leftarrow Q(s^i, a^i) + \alpha[r + \gamma \max_b Q(s'^i, ?) - Q(s^i, a^i)]$$

What value to use?

  - $s^i \leftarrow s'^i$

Joint Action Learners

- can we use ϵ-greedy over the Q values

- No, the Q function is multidimensional $Q(s, a^1, a^{\cdots}, a^n)$ and so the best value depends on the actions of the others…

  - Update Q estimate according to
    $$Q(s^i, a^i) \leftarrow Q(s^i, a^i) + \alpha[r + \gamma \max_b Q(s'^i, ?) - Q(s^i, a^i)]$$

  - Usually, the update is the estimate of the best value for next state, again, this depends on the actions of all other agents

Joint Action Learners

can we use ϵ-greedy over the Q values

- Now the Q function is multidimensional $Q(s, a^1, a^{\cdots}, a^n)$ and so the best value depends on the actions of the others…

Update Q estimate according to

$$Q(s^i, a^i) \leftarrow Q(s^i, a^i) + \alpha[r + \gamma \max_b Q(s'^i, ?) - Q(s^i, a^i)]$$

- Usually, the update is the estimate of the best value for next state, again, this depends on the actions of all other agents

→ For both processes we need to make some assumptions about the other agents

Learning in Multi-Agent Systems

Joint Action Learners

Assume the others are doing their best
Find a stable equilibria
Improve the worst case scenario

Joint Action Learners with Agent Modelling

Are they considering my preferences
Ignore others
Try to predict others

Learning in Multi-Agent Systems

Joint Action Learners

- Assume the others are doing their best
- Improve the worst case scenario

→minmaxQ

- Find a stable equilibria

→nashQ

Joint Action Learners with Agent Modelling

- Are they considering my preferences
- Try to predict others

→JAL-AM (extension of ficticious play for Markov Games)

Joint Action Learners

- A standard approach is to have each agent i maintain:
  - An **individual value function** $V_i(s)$
  - An **individual Q-function** $Q_i(s, a)$
    - Where s is the state and a is the joint actions

Joint Action Learners

- With the individual value function and individual Q-function, the **standard value iteration generalizes to Stochastic games** as follows:

$$V_i(s) \coloneqq C\big(Q_1(s,a), \dots, Q_n(s,a)\big), \forall s$$

$$Q_i(s,a) \coloneqq R_i(s,a) + \gamma \sum_{s' \epsilon S} P(s'|s,a) V_i(s'),\ \forall s, \forall a$$

- Where $\boldsymbol{C}$ **is a function that applies some solution concept** to the strategic game formed by $Q_1(s,a), \dots, Q_n(s,a)$

Joint Action Learners

- When **the transition model is not available**, a corresponding coupled update scheme can be derived in which

$$Q_i(s,a) \coloneqq R_i(s,a) + \gamma \sum_{s' \epsilon S} P(s'|s,a) V_i(s'),\ \forall s, \forall a$$

- is replaced by a Q-learning update, one per agent.

Joint Actions Learners

Where $\boldsymbol{C}$ **is a function that applies some solution concept** to the strategic game formed by $Q_1(s,a), \dots, Q_n(s,a)$

What solutions concepts can we have?

-assume worst case;

-assume nash-equilibria;

-...

Nash Q-learning

Definition 5 *Agent i's* Nash Q-function *is defined over* $(s, a^1, \ldots, a^n)$, *as the sum of Agent i's current reward plus its future rewards when all agents follow a joint Nash equilibrium strategy. That is,*

$$Q_*^i(s, a^1, \ldots, a^n) = r^i(s, a^1, \ldots, a^n) + \beta \sum_{s' \in S} p(s'|s, a^1, \ldots, a^n) v^i(s', \pi_*^1, \ldots, \pi_*^n), \tag{5}$$

where $(\pi_*^1, \ldots, \pi_*^n)$ *is the joint Nash equilibrium strategy,* $r^i(s, a^1, \ldots, a^n)$ *is agent i's one-period reward in state s and under joint action* $(a^1, \ldots, a^n)$, $v^i(s', \pi_*^1, \ldots, \pi_*^n)$ *is agent i's total discounted reward over infinite periods starting from state* s' *given that agents follow the equilibrium strategies.*

Nash Q-learning

Nash Q-learning algorithm

Initialize:

Let $t = 0$, get the initial state s_0.

Let the learning agent be indexed by i.

For all $s \in S$ and $a^j \in A^j$, $j = 1, \ldots, n$, let $Q_t^j(s, a^1, \ldots, a^n) = 0$.

Loop

Choose action a_t^i.

Observe $r_t^1, \ldots, r_t^n; a_t^1, \ldots, a_t^n$, and $s_{t+1} = s'$

Update Q_t^j for $j = 1, \ldots, n$

$$Q_{t+1}^j(s, a^1, \ldots, a^n) = (1 - \alpha_t) Q_t^j(s, a^1, \ldots, a^n) + \alpha_t [r_t^j + \beta NashQ_t^j(s')]$$

where $\alpha_t \in (0, 1)$ is the learning rate, and $NashQ_t^k(s')$ is defined in (7)

Let $t := t + 1$.

Hu, J. and Wellman, M. P. (2004). Nash Q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4:1039–1069.

Nash Q-learning

Q1=[0,0;
0,0]
Q2=[0,0;
0,0]

	Colaborate	***Defect***
Colaborate	0,0	0,0
Defect	0,0	0,0

- Play C, C

- Q1[C,C]=0+0.1(-1-0)=-0.1
- Q2[C,C]=0+0.1(-1-0)=-0.1

Prisoner 2

Prisoner 1

	Colaborate	***Defect***
Colaborate	-1, -1	-9, 0
Defect	0, -9	-6, -6

Nash Q-learning

Q1=[-0.1,0;
0,0]

Q2=[-0.1,0;
0,0]

	Colaborate	***Defect***
Colaborate	-0.1,-0.1	0,0
Defect	0,0	0,0

- Play D, D

- Q1[D,D]=0+0.1(-6)=-0.6
- Q2[D,D]=0+0.1(-6)=-0.6

Prisoner 2

Prisoner 1

	Colaborate	***Defect***
Colaborate	-1, -1	-9, 0
Defect	0, -9	-6, -6

Nash Q-learning

What is the next equilibria?

Q1=[-0.1,0;
0,-0.6]

Q2=[-0.1,0;
0,-0.6]

	Colaborate	***Defect***
Colaborate	-0.1,-0.1	0,0
Defect	0,0	-0.6,-0.6

- Play D, D

- Q1[D,D]=0+0.1(-6)=-0.6
- Q2[D,D]=0+0.1(-6)=-0.6

Prisoner 2

Prisoner 1		***Colaborate***	***Defect***
	Colaborate	-1, -1	-9, 0
	Defect	0, -9	-6, -6

Nash Q-learning

Q1=[-0.1,0;
0,-0.6]

Q2=[-0.1,0;
0,-0.6]

	Colaborate	***Defect***
Colaborate	-0.1,-0.1	0,0
Defect	0,0	-0.6,-0.6

- Play C, D

- Q1[C,D]=0+0.1(-9)=-0.9
- Q2[C,D]=0+0.1(0)=0

Prisoner 2

Prisoner 1

	Colaborate	***Defect***
Colaborate	-1, -1	-9, 0
Defect	0, -9	-6, -6

Nash Q-learning

Q1=[-0.1,-0.9;
0,-0.6]

Q2=[-0.1,0;
0,-0.6]

	Colaborate	***Defect***
Colaborate	-0.1,-0.1	-0.9,0
Defect	0,0	-0.6,-0.6

- Play D, C

- Q1[D,C]=0+0.1(0)=0
- Q2[D,C]=0+0.1(-9)=-0.9

Prisoner 2

Prisoner 1

	Colaborate	***Defect***
Colaborate	-1, -1	-9, 0
Defect	0, -9	-6, -6

Nash Q-learning

Did it converge?

Q1=[-0.1,-0.9;
0,-0.6]

Q2=[-0.1,0;
-0.9,-0.6]

	Colaborate	***Defect***
Colaborate	-0.1,-0.1	-0.9,0
Defect	0,-0.9	-0.6,-0.6

Prisoner 2

Prisoner 1	***Colaborate***	***Defect***
Colaborate	-1, -1	-9, 0
Defect	0, -9	-6, -6

minimaxQ

```
Initialize:
For all s in S, a in A, and o in O,
  Let Q[s,a,o] := 1
For all s in S,
  Let V[s] := 1
For all s in S, a in A,
  Let pi[s,a] := 1/|A|
Let alpha := 1.0
```

```
Choose an action:
With probability explor, return an action uniformly at random.
Otherwise, if current state is s,
  Return action a with probability pi[s,a].
```

```
Learn:
After receiving reward rew for moving from state s to s'
      via action a and opponent's action o,
  Let Q[s,a,o] := (1-alpha) * Q[s,a,o] + alpha * (rew + gamma * V[s'])
  Use linear programming to find pi[s,.] such that:
   pi[s,.]  := argmax{pi'[s,.], min{o', sum{a', pi[s,a'] * Q[s,a',o']}}}
  Let V[s] := min{o', sum{a', pi[s,a'] * Q[s,a',o']}}
  Let alpha := alpha * decay
```

Figure 1: The minimax-Q algorithm.

Only for zero-sum games

Coupled Learning

Correlated-Q Learning

MULTIQ(MarkovGame, γ, α, S, T)

Inputs	discount factor γ
	learning rate α
	decay schedule S
	total training time T
Output	action-value functions Q_i^*
Initialize	$s, a_1, \ldots, a_n$ and $Q_1, \ldots, Q_n$

for $t = 1$ to T

1. simulate actions $a_1, \ldots, a_n$ in state s
2. observe rewards $r_1, \ldots, r_n$ and next state s'
2. for $i = 1$ to n
   (a) compute $V_i(s')$
   (b) update $Q_i(s, a_1, \ldots, a_n)$
      i. $Q_i(s, a_1, \ldots, a_n) =$
   $(1-\alpha)Q_i(s, a_1, \ldots, a_n) + \alpha[r_i + \gamma V_i(s')]$
4. agents choose actions $a'_1, \ldots, a'_n$
5. $s = s'$, $a_1 = a'_1, \ldots, a_n = a'_n$
6. decay α according to S

A **correlated equilibrium** is a generalization of a mixed-strategy Nash equilibrium where the mixed strategies of the agents can be correlated

Greenwald, A. and Hall, K. (2003). Correlated-Q learning. In *Proc. 20th Int. Conf. on Machine Learning*, Washington, DC, USA

Coupled Learning

- The algorithms require that **a learning agent must know**:
  - The actions of other agents
  - The rewards of the other agents
- **Complexity**:
  - The learning agent has to maintain n Q-tables, where each table has the size $|S| \times |A|^n$ (i.e., exponential in the number of agents)
  - The running time of the function that applies the solution concept can also be exponential

Outline

- Stochastic games
- Learning in repeated games
- Independent learning
- **Multiagent learning**
  - Joint-Action Learners
  - Agent-Modelling

Fictitious Play

- Fictitious play is an **instance of model-based** learning
  - The learning agent **explicitly maintains beliefs about the opponent's actions**
- In fictitious play, the **learning agent is oblivious to the payoffs obtained by other agents**

Learning in Repeated Games

- How can an agent **learn to play a repeated game**?
  - A famous learning algorithm is called **Fictitious play**
    - It was actually not proposed initially as a learning model
    - Originally, it was used as an iterative method for computing Nash equilibria in zero-sum games

Fictitious Play

Fictitious play algorithm:

Initialize beliefs about the opponent's action

repeat

- Play a best response to the assessed action of the opponent
- Observe the opponent's actual play and update beliefs accordingly

Fictitious Play

- In Fictitious play, we assume:
  - the **agent does not know the payoffs of the other agents**
  - the **agent only needs to know his own payoff matrix**
    - i.e., the payoff he would get in each joint action profile, whether or not encountered in the past
- Also, in fictitious play, **the learning agent believes that his opponent is playing the mixed strategy** given by the empirical distribution of the opponent's previous actions

Fictitious Play

- Let:
  - A be the set of the opponent's actions
  - $w(a)$ be the number of times that the opponent has played action a, for every $a \in A$

- Hence, the **learning agent assesses the probability of a in the opponent's mixed strategy** as:

$$P(a) = \frac{w(a)}{\sum_{a' \in A} w(a')}$$

Fictitious Play

- For example, in a **repeated Prisoner's Dilemma**, if the opponent has selected the following actions in the first five games:
  - *Confess, Confess, Not Confess, Confess, Not Confess*

- Before the sixth game, the learning agent has $w(Confess) = 3$ and $w(Not\ Confess) = 2$ and thus
  - The agent thus assumes that the opponent is playing the mixed strategy $(0.6, 0.4)$

		Prisoner 2	
		Not confess	***Confess***
Prisoner 1	***Not confess***	-1, -1	-9, 0
	Confess	0, -9	-6, -6

Fictitious Play

- Note that **we can represent an opponent's beliefs** with either a **probability measure** or with the **set of counts** $(w(a_1), \ldots, w(a_k))$

- Note that **different versions of fictitious play exist** depending on the **tie-breaking method** used to select an action when **there is more than one best response**

  - In general, the **tie-breaking rule chosen has little effect on the results of fictitious play**

Fictitious Play

- However, fictitious play is **very sensitive to the players' initial beliefs**
  - The initial beliefs can be interpreted as action counts that were observed before the start of the game
  - These **initial beliefs can have a radical impact on the learning process**
- Note that **one must pick some nonempty prior belief** for each agent
  - the prior beliefs cannot be (0, . . . , 0) since this does not define a meaningful mixed strategy

Fictitious Play

- **Example**:
  - Repeated game of Matching Pennies

		Agent 2	
		Heads	***Tails***
Agent 1	***Head***	1, -1	-1, 1
	Tails	-1, 1	1, -1

Fictitious Play

- Repeated game of Matching Pennies:

	Heads	***Tails***
Head	1, -1	-1, 1
Tails	-1, 1	1, -1

Round	1's action	2's action	1's beliefs about 2	2's beliefs about 1
0			(1.5,2)	(2,1.5)

Fictitious Play

- Repeated game of Matching Pennies:

	Heads	***Tails***
Head	1, -1	-1, 1
Tails	-1, 1	1, -1

Round	1's action	2's action	1's beliefs about 2	2's beliefs about 1
0			(1.5,2)	(2,1.5)
1	T	T	(1.5,**3**)	(2,**2.5**)

Update beliefs

Fictitious Play

- Repeated game of Matching Pennies:

	Heads	***Tails***
Head	1, -1	-1, 1
Tails	-1, 1	1, -1

Round	1's action	2's action	1's beliefs about 2	2's beliefs about 1
0			(1.5,2)	(2,1.5)
1	T	T	(1.5,3)	(2,2.5)
2	T	H	(**2.5**,3)	(2,**3.5**)

Update beliefs

Fictitious Play

- Repeated game of Matching Pennies:

	Heads	***Tails***
Head	1, -1	-1, 1
Tails	-1, 1	1, -1

Round	1's action	2's action	1's beliefs about 2	2's beliefs about 1
0			(1.5,2)	(2,1.5)
1	T	T	(1.5,3)	(2,2.5)
2	T	H	(2.5,3)	(2,3.5)
3	T	H	(**3.5**,3)	(2,**4.5**)

Update beliefs

Fictitious Play

- Repeated game of Matching Pennies:

	Heads	***Tails***
Head	1, -1	-1, 1
Tails	-1, 1	1, -1

Round	1's action	2's action	1's beliefs about 2	2's beliefs about 1
0			(1.5,2)	(2,1.5)
1	T	T	(1.5,3)	(2,2.5)
2	T	H	(2.5,3)	(2,3.5)
3	T	H	(3.5,3)	(2,4.5)
4	H	H	(**4.5**,3)	(**3**,4.5)

Update beliefs

Fictitious Play

- Repeated game of Matching Pennies:

	Heads	***Tails***
Head	1, -1	-1, 1
Tails	-1, 1	1, -1

Round	1's action	2's action	1's beliefs about 2	2's beliefs about 1
0			(1.5,2)	(2,1.5)
1	T	T	(1.5,3)	(2,2.5)
2	T	H	(2.5,3)	(2,3.5)
3	T	H	(3.5,3)	(2,4.5)
4	H	H	(4.5,3)	(3,4.5)
5	H	H	(**5.5**,3)	(**4**,4.5)

Update beliefs

Fictitious Play

- Repeated game of Matching Pennies:

	Heads	*Tails*
Head	1, -1	-1, 1
Tails	-1, 1	1, -1

Round	1's action	2's action	1's beliefs about 2	2's beliefs about 1
0			(1.5,2)	(2,1.5)
1	T	T	(1.5,3)	(2,2.5)
2	T	H	(2.5,3)	(2,3.5)
3	T	H	(3.5,3)	(2,4.5)
4	H	H	(4.5,3)	(3,4.5)
5	H	H	(5.5,3)	(4,4.5)
6	H	H	(**6.5**,3)	(**5**,4.5)

Update beliefs

Fictitious Play

- Repeated game of Matching Pennies:

	Heads	*Tails*
Head	1, -1	-1, 1
Tails	-1, 1	1, -1

Round	1's action	2's action	1's beliefs about 2	2's beliefs about 1
0			(1.5,2)	(2,1.5)
1	T	T	(1.5,3)	(2,2.5)
2	T	H	(2.5,3)	(2,3.5)
3	T	H	(3.5,3)	(2,4.5)
4	H	H	(4.5,3)	(3,4.5)
5	H	H	(5.5,3)	(4,4.5)
6	H	H	(6.5,3)	(5,4.5)
7	H	T	(6.5,**4**)	(**6**,4.5)
…	…	…	…	…

Update beliefs

Fictitious Play

- As we can see, each player ends up alternating back and forth between playing heads and tails

- In fact, as the number of rounds tends to infinity, the empirical distribution of each player will converge to $(0.5, 0.5)$

- If we take this distribution to be the mixed strategy of each player
  - **the play converges to the unique Nash equilibrium** of the normal form game
  - In the repeated Matching Pennies, each player plays the mixed strategy (0.5, 0.5)

Joint Action Learning with Agent Modelling (JAL-AM)

How to expand ficticious play for stochastic games?

$$\hat{\pi}_j(a_j \mid s) = \frac{C(s, a_j)}{\sum_{a'_j} C(s, a'_j)}.$$

Is it good to use this equation? If the policy changes how fast it will converge?

$$AV_i(s, a_i) = \sum_{a_{-i} \in A_{-i}} Q_i(s, \langle a_i, a_{-i} \rangle) \prod_{j \neq i} \hat{\pi}_j(a_j \mid s)$$

https://www.marl-book.com/

Joint Action Learning with Agent Modelling (JAL-AM)

Algorithm 8 Joint Action Learning with Agent Modelling (JAL-AM)

// Algorithm controls agent i

Initialise:

  $Q_i(s,a) = 0$ for all $s \in S, a \in A$

  Agent models $\hat{\pi}_j(a_j \mid s) = \frac{1}{|A_j|}$ for all $j \neq i, a_j \in A_j, s \in S$

Repeat for every episode:

for $t = 0, 1, 2, \ldots$ **do**

  Observe current state s^t

  With probability ϵ: choose random action a_i^t

  Else: choose best-response action $a_i^t \in \arg\max_{a_i} AV_i(s^t, a_i)$

  Observe joint action $a^t = (a_1^t, \ldots, a_n^t)$, reward r_i^t, next state s^{t+1}

  for all $j \neq i$ **do**

    Update agent model $\hat{\pi}_j$ with new observations (e.g. (s^t, a_j^t))

  $Q_i(s^t, a^t) \leftarrow Q_i(s^t, a^t) + \alpha \left[r_i^t + \gamma \max_{a_i'} AV_i(s^{t+1}, a_i') - Q_i(s^t, a^t) \right]$

https://www.marl-book.com/

Are Equilibria Good Predictor of Human Behavior?

- In prisioner’s dilema do people always end in the Nash equilibria?
- What is your expectation?

Are Equilibria Good Predictor of Human Behavior?

- What is the equilibria in the following game?

- What would you expect to happen with 2 people?

Are Equilibria Good Predictor of Human Behavior?

- Consider two strategies in PD
  - Nash-Equilibria
  - Tit-for-Tat – Defect after defect
  - Tit-for-2Tats – Defect after 2 defects

- What would you expect to happen with 2 people?

Are Equilibria Good Predictor of Human Behavior?

Tit-for-TaT

Tit-for-2Tats

Behavior Profiles

- The behavior of people tend to be more pro-social than what pure utility based approaches predict.

- Over many interactions we might start to expect specific behaviors of people.

- Ficticious play cannot address profiles of people and always acts as with the best response for the mixed strategy observed up to the moment.

→ How to act when we know that there are several profiles of people?

Rational Learning

Let's assume that there are two profiles of people, nice and not nice.

Nice people collaborate 90% of the time

$p(C|N)=0.9$

Not nice people collaborate only 10% of the time

$p(C|\tilde{N})=0.1$

Rational Learning

If during play one player collaborates what is the probability that they are Nice?

$$p(N|C) = \frac{p(C|N)p(N)}{\sum_b p(C|b)p(b)} \propto p(C|N)p(N)$$

$$p(N|C) \propto 0.9 \times 0.5$$
$$p(\tilde{N}|C) \propto 0.1 \times 0.5$$
$$p(N|C) = \frac{p(N|C)}{p(N|C) + p(\tilde{N}|C)} = 0.9$$

Rational Learning

If during play one player plays C,D,C, what is the probability that they are Nice?

$$p(N|C,D,C) \propto p(C,D,C|N)p(N)$$
$$\propto p(C|N)p(D|N)p(C|N)p(N)$$
$$\propto 0.9 \times 0.1 \times 0.9 \times 0.5 = 0.0405$$
$$p(N|C,D,C) = \frac{0.0405}{0.0405 + 0.0045} = 0.9$$

Assuming independ e

Rational Learning

If during play one player plays C,D,C, what is the probability that they are Nice?

$$p(N|C,D,C) \propto p(C,D,C|N)p(N)$$
$$\propto p(C|N)p(D|N)p(C|N)p(N)$$

The rule can be applied incrementally and at each step the prior is the posterior of the previous step.

Rational Learning

How to compute the best response?

In ficticious play we assume that they are playing the mixed strategy that we computed up to the current moment. Now we have probabilities over behaviors.

Again a Bayesian risk analysis can be used.

Again U_1(a_1,a_2) it is the utility for agent 1 when the players play a_1,a_2

p(action = a|profile = b) it is the probability of playing a if the player profile is b

Rational Learning

How to compute the best response?

Again a Bayesian risk analysis can be used.

Again U_1(a_1,a_2) it is the utility for agent 1 when the players play a_1,a_2

p(action = a|profile = b) it is the probability of playing a if the player profile is b

Utility(Cooperate) =
$\sum_b \sum_a U(C,a)$p(action = a|profile = b)$p(profile = b)$

Other Learning Approaches

Policy gradient

Instead of learning the value and then choose the actions based on a given criteria, the values obtained are used to change directly the policy/strategy (using gradient ascent).

Policy-Based Learning

- Assume the payoffs for each agent

$$\mathcal{R}_i = \begin{bmatrix} r_{1,1} & r_{1,2} \\ r_{2,1} & r_{2,2} \end{bmatrix} \qquad \mathcal{R}_j = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}$$

- And the following strategies (one advantage is that we can reason directly with mixed strategies)

$$\pi_i = (\alpha, 1-\alpha) \qquad \pi_j = (\beta, 1-\beta), \qquad \alpha, \beta \in [0, 1]$$

- What is the expected payoff for each agent?

Policy-Based Learning

$$\mathcal{R}_i = \begin{bmatrix} r_{1,1} & r_{1,2} \\ r_{2,1} & r_{2,2} \end{bmatrix} \qquad \mathcal{R}_j = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}$$

$$\pi_i = (\alpha, 1-\alpha) \qquad \pi_j = (\beta, 1-\beta), \qquad \alpha, \beta \in [0, 1]$$

The expected payoff for each is:

$$U_i(\alpha, \beta) = \alpha\beta r_{1,1} + \alpha(1-\beta)r_{1,2} + (1-\alpha)\beta r_{2,1} + (1-\alpha)(1-\beta)r_{2,2}$$

$$= \alpha\beta u + \alpha(r_{1,2} - r_{2,2}) + \beta(r_{2,1} - r_{2,2}) + r_{2,2}$$

$$U_j(\alpha, \beta) = \alpha\beta c_{1,1} + \alpha(1-\beta)c_{1,2} + (1-\alpha)\beta c_{2,1} + (1-\alpha)(1-\beta)c_{2,2}$$

$$= \alpha\beta u' + \alpha(c_{1,2} - c_{2,2}) + \beta(c_{2,1} - c_{2,2}) + c_{2,2}$$

$$u = r_{1,1} - r_{1,2} - r_{2,1} + r_{2,2}$$

$$u' = c_{1,1} - c_{1,2} - c_{2,1} + c_{2,2}.$$

Policy-Based Learning

The expected payoff for each is:

$$U_i(\alpha, \beta) = \alpha\beta r_{1,1} + \alpha(1-\beta) r_{1,2} + (1-\alpha)\beta r_{2,1} + (1-\alpha)(1-\beta) r_{2,2}$$

$$= \alpha\beta u + \alpha(r_{1,2} - r_{2,2}) + \beta(r_{2,1} - r_{2,2}) + r_{2,2}$$

$$u = r_{1,1} - r_{1,2} - r_{2,1} + r_{2,2}$$

The gradient update:

$$\alpha^{k+1} = \alpha^k + \kappa \frac{\partial U_i(\alpha^k, \beta^k)}{\partial \alpha^k}$$

Where

$$\frac{\partial U_i(\alpha, \beta)}{\partial \alpha} = \beta u + (r_{1,2} - r_{2,2})$$

Policy-Based Learning

The expected payoff for each is:

$$U_i(\alpha, \beta) = \alpha\beta r_{1,1} + \alpha(1-\beta) r_{1,2} + (1-\alpha)\beta r_{2,1} + (1$$

$$= \alpha\beta u + \alpha(r_{1,2} - r_{2,2}) + \beta(r_{2,1} -$$

$$u = r_{1,1} - r_{1,2} - r_2$$

To do this we need to know all the utility matrices and the policy of the other agent at each step...

The gradient update:

$$\alpha^{k+1} = \alpha^k + \kappa$$

Where

$$\frac{\partial U_i(\alpha, \beta)}{\partial \alpha} = \beta u + (r_{1,2} - r_{2,2})$$

Generalised Infinitesimal Gradient Ascent

Given a policy π_i for agent i and an action a_j for agent j, the expected reward for agent i against action a_j is

$$U_i(\pi_i, a_j) = \sum_{a_i \in A_i} \pi_i(a_i) \mathcal{R}_i(a_i, a_j). \tag{6.46}$$

Therefore, the gradient of this expected reward with respect to policy π_i is simply the vector of rewards for each of agent i's available actions $1, 2, 3...$,

$$\nabla_{\pi_i} U_i(\pi_i, a_j) = \left[\frac{\partial U_i(\pi_i, a_j)}{\partial \pi_i(1)}, \quad \frac{\partial U_i(\pi_i, a_j)}{\partial \pi_i(2)}, \quad \frac{\partial U_i(\pi_i, a_j)}{\partial \pi_i(3)}, \quad \ldots \right] \tag{6.47}$$

$$= \left[\mathcal{R}_i(1, a_j), \quad \mathcal{R}_i(2, a_j), \quad \mathcal{R}_i(3, a_j), \quad \ldots \right]. \tag{6.48}$$

Given policy π_i^k and observed action a_j^k in episode k, GIGA updates π_i^k via two steps:

$$\begin{aligned} &(1) \quad \tilde{\pi}_i^{k+1} \leftarrow \pi_i^k + \kappa^k \nabla_{\pi_i^k} U_i(\pi_i^k, a_j^k) \\ &(2) \quad \pi_i^{k+1} \leftarrow P(\tilde{\pi}_i^{k+1}) \end{aligned} \tag{6.49}$$

Generalised Infinitesimal Gradient Ascent

```
    pols =[[],[]]
    pols[0] = np.ones(n0)/n0
    pols[1] = np.ones(n1)/n1

    A = [-1,-1]
    for ii in range(200):
        #select action for each agent, $\epsilon-greedy$ might
work
        A[0] = np.random.choice(n0, p=pols[0])
        A[1] = np.random.choice(n1, p=pols[1])

        r = get_value(g,A[0],A[1])

        d0 = g.payoff_matrices[0][:,A[1]]
        d1 = g.payoff_matrices[1][A[0],:]

        pols[0] = np.clip(pols[0] + 0.01 * d0,0,1)
        pols[1] = np.clip(pols[1] + 0.01 * d1,0,1)
        pols[0] *= 1/np.sum(pols[0])
        pols[1] *= 1/np.sum(pols[1])
```

Generalised Infinitesimal Gradient Ascent

Generalised Infinitesimal Gradient Ascent

Learning in Multi-Agent Systems

What should the criteria be?

- Assume the others are doing their best
- Are they considering my preferences
- Find a stable equilibria
- Improve the worst case scenario
- Ignore others
- Try to predict others

https://openspiel.readthedocs.io/en/latest/algorithms.html

AlphaZero (C++/LibTorch)	MARL	Silver et al. '18	
AlphaZero (Python/TF)	MARL	Silver et al. '18	
Correlated Q-Learning	MARL	Greenwald & Hall '03	~
Asymmetric Q-Learning	MARL	Kononen '04	~
Deep CFR	MARL	Brown et al. '18	
DiCE: The Infinitely Differentiable Monte-Carlo Estimator (LOLA-DiCE)	MARL	Foerster, Farquhar, Al-Shedivat et al. '18	~
Exploitability Descent (ED)	MARL	Lockhart et al. '19	
(Extensive-form) Fictitious Play (XFP)	MARL	Heinrich, Lanctot, & Silver '15	
Learning with Opponent-Learning Awareness (LOLA)	MARL	Foerster, Chen, Al-Shedivat, et al. '18	~
Nash Q-Learning	MARL	Hu & Wellman '03	~
Neural Fictitious Self-Play (NFSP)	MARL	Heinrich & Silver '16	
Neural Replicator Dynamics (NeuRD)	MARL	Omidshafiei, Hennes, Morrill, et al. '19	X
Regret Policy Gradients (RPG, RMPG)	MARL	Srinivasan, Lanctot, et al. '18	
Policy-Space Response Oracles (PSRO)	MARL	Lanctot et al. '17	
Q-based ("all-actions") Policy Gradient (QPG)	MARL	Srinivasan, Lanctot, et al. '18	
Regularized Nash Dynamics (R-NaD)	MARL	Perolat, De Vylder, et al. '22	
Regression CFR (RCFR)	MARL	Waugh et al. '15, Morrill '16	
Rectified Nash Response (PSRO_rn)	MARL	Balduzzi et al. '19	~

Summary:
Learning in Markov Games

Reference: www.marl-book.com
Chapter 6 – 6.1, 6.2, 6.3, 6.4

Multiagent decision making and Auctions

AASMA 2023

Alberto Sardinha
jose.alberto.sardinha@tecnico.ulisboa.pt
Rui Prada
rui.prada@tecnico.ulisboa.pt

Outline

- **Introduction to auctions**
- Canonical auctions
- Bidding in first-price auctions
- Bidding in second-price auctions

Introduction

Introduction

Introduction

Introduction

Introduction

Auctions

- Auctions are a **mechanism for allocating resources among self-interested agents**
  - Normally scarce resources

- Widely used to:
  - Sell art
  - Sell public companies (privatization)
  - Sell or buy stocks
  - Sell used goods (e.g., eBay)
  - Procure parts
  - Etc.

Outline

- Introduction to auctions
- **Canonical auctions**
- Bidding in first-price auctions
- Bidding in second-price auctions

Canonical Auctions

- English auction
- Dutch auction
- First-price auction
- Second-price auction

English Auction

- An **English auction** is an open-outcry ascending auction that proceeds as follows:
  - The **auctioneer starts the bidding** at some starting price (reserve price)
  - **Bidders** then **shout out ascending prices**
  - At any given moment, the **highest bidder** is considered to have the **standing bid**

English Auction

- An **English auction** is an open-outcry ascending auction that proceeds as follows:
  - The **standing bid becomes the winner** if **no competing bidder challenges the standing bid** within a given time frame
    - And the **item is sold to the highest bidder** at a **price equal to their bid**

English Auction

- **Example**

English Auction

- Interesting properties/facts of the **English auctions**:
  - Every bidder knows the number of bidders in the auction
  - The bids are public
  - Bidders can submit several bids
  - This type of auction is commonly used for selling art, antiques, wine, etc.

Dutch Auction

- A **Dutch auction** is an open-outcry descending auction that proceeds as follows:
  - The auctioneer starts a clock at some high asking price
  - The price lowers at each time step until a bidder accepts the current ask price (by shouting)

Dutch Auction

- **Example:**
  - A farmer wants to sell a basket of apples and uses a Dutch auction
  - The starting bid is $150
  - If nobody accepts the initial bid, the farmer (auctioneer) successively reduces the price in increments of $10 after 5 seconds:
    - t = 0 : ask price = $150
    - t = 5 : ask price = $140
    - t = 10 : ask price = $130
    - …

Dutch Auction

- **Example**:
  - A particular bidder is the first to shout out that he wants to buy the item when the price reaches $40
    - Note that the bidder feels that price is acceptable and that someone else might bid soon
    - The bidder pays $40 for the basket of apples

Dutch Auction

- Interesting properties/facts of the **Dutch auctions**:
  - Every bidder knows the number of bidders in the auction
  - The bid is public (and only one bid is submitted)
  - This type of auction is commonly used for selling flowers, fresh produce, tobacco, etc

First-Price Auction

- A **first-price sealed-bid auction** proceeds as follows:
  - All bidders submit sealed bids simultaneously
    - No bidder knows the bids of the other bidders
  - The highest bidder wins and pays the submitted price

First-Price Auction

- **Example**:
  - Bidder 1 submits a sealed bid of $100
  - Bidder 2 submits a sealed bid of $80
  - Bidder 3 submits a sealed bid of $95
  - Bidder 1 wins and pays $100 for the item

First-Price Auction

- Interesting properties/facts of the **first-price auctions**:
  - Every bidder knows the number of bidders in the auction
  - The bids are private
  - Bidders can only submit one bid
  - This type of auction is commonly used for privatization of public companies, selling concessions, etc

Second-Price Auction

- A **second-price sealed-bid auction** proceeds as follows:
  - All bidders submit sealed bids simultaneously
    - No bidder knows the bids of the other bidders
  - The highest bidder wins and pays the second highest bid

Second-Price Auction

- **Example**:
  - Bidder 1 submits a sealed bid of $100
  - Bidder 2 submits a sealed bid of $80
  - Bidder 3 submits a sealed bid of $95
  - Bidder 1 wins and pays $95 for the item (and not $100!)

Second-Price Auction

- Interesting properties/facts of the **second-price auctions**:
  - Every bidder knows the number of bidders in the auction
  - The bids are private
  - Bidders can only submit one bid
  - Bidders can be invited/selected to the auction
  - It is used in digital ads tech (e.g., Google and Facebook)
  - It has very interesting theoretical results

Outline

- Introduction to auctions
- Canonical auctions
- **Bidding in first-price auctions**
- Bidding in second-price auctions
- Revenue equivalence

Bidding in First-Price Auctions

- How do agents bid in a first-price auction?
  - An agent has an incentive to bid less than its true valuation
    - For instance, if the agent thinks the value of a good is $10 then he might want to bid $8
      - In a first price auction, if the agent bids $8 and wins, then he pays $8 and makes a profit equal to $2 (i.e., profit = $10 - $8 = $2)

Bidding in First-Price Auctions

- How do agents bid in a first-price auction?
  - The tradeoff in the bidding decision:
    - Probability of winning
      - lower bid → probability of winning is lower
      - higher bid → probability of winning is higher
    - Amount paid when winning
      - lower bid → higher profit
      - higher bid → lower profit

Bidding in First-Price Auctions

- How do agents bid in a first-price auction?
  - Bidders do not have a dominant strategy
    - strategy of player i depends on the strategy of other players

Bidding in First-Price Auctions

- How do agents bid in a first-price auction?
  - **Theorem**: In a first-price sealed-bid auction with:
    - Two risk-neutral bidders (i.e., agent 1 and agent 2)
    - The valuations v_1 and v_2 are i.i.d. and drawn from $U(0,1)$
    - Hence:
      - Agent 1 bids $\frac{1}{2}v_1$
      - Agent 2 bids $\frac{1}{2}v_2$
      - And $\left(\frac{1}{2}v_1, \frac{1}{2}v_2\right)$ is a Bayesian-Nash equilibrium

Bidding in First-Price Auctions

- **Proof:**
  - Let us assume bidder 2 bids $b_2 = \frac{1}{2} v_2$
    - Where v_2 is the value of the object from bidder 2's perspective
  - We now analyse bidder 1's optimal decision (best response) :
    - The optimal decision is is to maximize the expected profit:

$$\max_{b_1} \mathbb{E}[u_1]$$

Bidding in First-Price Auctions

- **Proof:**
  - We now analyse bidder 1's optimal decision (best response):
    - bidder 1 wins the auction when $b_2 < b_1$, hence $v_2 < 2b_1$ with profit $u_1 = v_1 - b_1$
    - bidder 1 looses the auction when $b_1 < b_2$, hence $v_2 > 2b_1$ with profit $u_1 = 0$
    - Hence, $\mathbb{E}[u_1] = P(win|b_1)(v_1 - b_1) + P(loose|b_1)0$

Bidding in First-Price Auctions

- **Proof:**
  - We now analyse bidder 1's optimal decision (best response):
    - $\mathbb{E}[u_1] = P(win|b_1)(v_1 - b_1) + P(loose|b_1)\,0$
    - $\mathbb{E}[u_1] = P(win|b_1)(v_1 - b_1)$

Bidding in First-Price Auctions

- **Proof:**
  - We now analyse bidder 1's optimal decision (best response):
    - $\mathbb{E}[u_1] = P(win|b_1)(v_1 - b_1)$
    - $\mathbb{E}[u_1] = P(b_2 < b_1)(v_1 - b_1)$
  - Substituting $v_2 < 2b_1$ for $b_2 < b_1$:
    - $\mathbb{E}[u_1] = P(v_2 < 2b_1)(v_1 - b_1)$

Bidding in First-Price Auctions

- **Proof:**
  - We now analyse bidder 1's optimal decision (best response):
    - $\mathbb{E}[u_1] = P(v_2 < 2b_1)(v_1 - b_1)$
    - $\mathbb{E}[u_1] = F_{v2(}2b_1)(v_1 - b_1)$

RECALL:
The ***cumulative distribution function*** **(CDF)** of a real-valued random variable X is the function given by:

$$F_X(x) = P(X \leq x)$$

Bidding in First-Price Auctions

- **Proof:**
  - We now analyse bidder 1's optimal decision (best response):
    - $\mathbb{E}[u_1] = F_{v2}(2b_1)(v_1 - b_1)$
    - $\mathbb{E}[u_1] = \int_0^{2b_1} dv_2 \ (v_1 - b_1)$

RECALL:
The CDF can be expressed as the integral of its probability density function:

$$F_X(x) = \int_{-\infty}^{x} f_X(t)dt$$

Probability density function
(uniform distribution)

Cumulative distribution function
(uniform distribution)

Bidding in First-Price Auctions

- **Proof:**
  - We now analyse bidder 1's optimal decision (best response):
    - $\mathbb{E}[u_1] = \int_0^{2b_1} dv_2 \ (v_1 - b_1)$
    - $\mathbb{E}[u_1] = (v_2 + k)|_0^{2b_1}(v_1 - b_1)$
    - $\mathbb{E}[u_1] = 2b_1(v_1 - b_1) = 2v_1b_1 - 2b_1^2$

Bidding in First-Price Auctions

- **Proof:**
  - We now analyse bidder 1's optimal decision (best response):
    - Recall that we want to maximize the expected profit:

$$\max_{b_1} \mathbb{E}[u_1]$$

    - Hence, the first order condition is $\frac{\partial \mathbb{E}[u_1]}{\partial b_1} = 0$

Bidding in First-Price Auctions

- **Proof:**
  - We now analyse bidder 1's optimal decision (best response):
    - $\frac{\partial \mathbb{E}[u_1]}{\partial b_1} = \frac{\partial}{\partial b_1} 2v_1 b_1 - 2b_1^2 = 0$
    - $2v_1 - 4b_1 = 0$
    - $b_1 = \frac{v_1}{2}$

Bidding in First-Price Auctions

- **Proof:**
  - Let us assume bidder 1 bids $b_1 = \frac{1}{2} v_1$
    - Where v_1 is the value of the object from bidder 1's perspective
  - We now analyse bidder 2's optimal decision (best response) :
    - The optimal decision is is to maximize the expected profit:

$$\max_{b_2} \mathbb{E}[u_2]$$

    - And following the same steps in the previous slides:

$$b_2 = \frac{v_2}{2}$$

    - Hence, $\left(\frac{1}{2} v_1, \frac{1}{2} v_2\right)$ is a Bayesian-Nash equilibrium

Bidding in First-Price Auctions

- How do agents bid in a first-price auction?
  - **Theorem**: In a first-price sealed-bid auction with:
    - N risk-neutral bidders
    - The valuations v_i are i.i.d. and drawn from $U(0,1)$
    - Hence:
      - $\left(\frac{N-1}{N}v_1, \frac{N-1}{N}v_2, \dots, \frac{N-1}{N}v_N\right)$ is a Bayesian-Nash equilibrium

Outline

- Introduction to auctions
- Canonical auctions
- Bidding in first-price auctions
- **Bidding in second-price auctions**
- Revenue equivalence

Bidding in Second-Price Auctions

- How do agents bid in a second-price auction?
  - **Theorem**: In a second-price sealed-bid auction with:
    - N risk-neutral bidders
    - The valuations v_i
    - Hence:
      - $(v_1, v_2, \dots, v_N)$ is a Nash equilibrium

Bidding in Second-Price Auctions

- Consider the **losers' strategies**:
  - The **profit/payoff of a loosing bid is equal to zero in the NE** (i.e., $b_i = v_i$)
  - **Reducing their bids does not change the profit** because they **still loose and do not pay anything**
    - profit/payoff is still equal to zero

Bidding in Second-Price Auctions

- Consider the **losers' strategies**:
  - Increasing their bids above their valuation may or may not change the profit:
    - **If they still loose with the increased bid, they still do not pay anything**
      - profit/payoff is still equal to zero

Bidding in Second-Price Auctions

- Consider the **losers' strategies**:
  - Increasing their bids above their valuation may or may not change the profit:
    - On the other hand, **a loser increasing his bid to a winning price gives him the good, but at a price higher than the maximum he was willing to pay**
      - profit/payoff is negative

Bidding in Second-Price Auctions

- Consider the **losers' strategies**:
  - Consequently, losers have no incentives to deviate from the NE
    - i.e., losers have no incentives to change their bids

Bidding in Second-Price Auctions

- Consider the **winner's strategy**:
  - **Increasing his bid does not change anything**
    - He will still win and continue to pay the price of the second highest bid
      - profit/payoff is the same

Bidding in Second-Price Auctions

- Consider the **winner's strategy**:
  - Decreasing the bid can only hurt him
    - If **the decreased bid stays above the second highest bid, he still wins and still pays the second highest bid**
      - profit/payoff is the same

Bidding in Second-Price Auctions

- Consider the **winner's strategy**:
  - Decreasing the bid can only hurt him
    - If **the decreased bid drops below the second highest bid, he now looses the auction**
      - profit/payoff is equal to zero

Bidding in Second-Price Auctions

- Consider the **winner's strategy**:
  - Consequently, the winner has no incentives to deviate from the NE
    - i.e., winner has no incentive to change his bid

Outline

- Introduction to auctions
- Canonical auctions
- Bidding in first-price auctions
- Bidding in second-price auctions
- **Revenue equivalence**

Revenue Equivalence

Which auction should an auctioneer choose?

Revenue Equivalence

Which auction should an auctioneer choose?

To some extent, it does not matter...

Revenue Equivalence

- **Theorem** (**Revenue Equivalence Theorem**): *Assume that each of n risk-neutral agents has an independent private valuation for a single good at auction, drawn from a common cumulative distribution $F(v)$ that is strictly increasing and atomless on $[\underline{v}, \overline{v}]$. Then any auction mechanism in which*

  - *the good will be allocated to the agent with the highest valuation; and*

  - *any agent with valuation $\underline{v}$ has an expected utility of zero;*

yields the same expected revenue, and hence results in any bidder with valuation v making the same expected payment.

Revenue Equivalence

- Can we use **Revenue Equivalence Theorem** with the first-price and second-price auctions?

YES!

- Why?
  - The first-price and second-price auctions are **symmetric games** and **every symmetric game has a symmetric equilibrium.** In addition, a symmetric equilibrium has the following property:

    higher bid ⇔ higher valuation

  - Hence, the good will be allocated to the agent with the highest valuation
  - And any agent with valuation $\underline{v}$ has an expected utility of zero

Revenue Equivalence

- We will use $\boldsymbol{k^{th}}$ **order statistic of a distribution** to analyze the revenue equivalence in first-price and second-price auctions

- $\boldsymbol{k^{th}}$ **order statistic of a distribution:** the expected value of the kth-largest of n draws

- For n i.i.d. draws from a uniform distribution $[0, v_{max}]$, the kth order statistic is:

$$\frac{n+1-k}{n+1} v_{max}$$

Revenue Equivalence

- **First-price auction**:
  - Recall that the winner pays the largest bid
  - However, following the **Revenue Equivalence Theorem**, the winning bidder in a first-price auction must bid his expected payment conditional on being the winner of a second-price auction

Revenue Equivalence

- The winning bidder in a first-price auction must bid his expected payment conditional on being the winner of a second-price auction
  - **If bidder i's valuation v_i is the highest**, there are then $n - 1$ other valuations drawn from the uniform distribution on $[0, v_i]$
  - Hence, the expected value of the second-highest valuation (bid) is the first-order statistic of $n - 1$ draws from $[0, v_i]$:

$$\frac{(n-1)+1-1}{(n-1)+1} v_i = \frac{n-1}{n} v_i$$

  - This provides a basis for our earlier claim about n-bidder first-price auctions

Revenue Equivalence

- This provides a basis for our earlier claim about n-bidder first-price auctions
  - However, we would still **have to check that this is an equilibrium**!
  - The revenue equivalence theorem does not say that every revenue-equivalent strategy profile is an equilibrium!

Thank You

jose.alberto.sardinha@tecnico.ulisboa.pt
rui.prada@tecnico.ulisboa.pt

Social Choice / Voting

AASMA 2025

Manuel Lopes
manuel.lopes@tecnico.ulisboa.pt

Voting

- How to make a decision that respects everyone’s wishes?

  - -> consensus?
  - -> majority?
  - -> democratic?

Problems

- Wishes might be incompatible
- When making a choice
- When electing representatives problems with discretizations

Social Choice

Our setting now:

- a set of outcomes
- agents have preferences across them
- for the moment, we won't consider incentive issues:
  - center knows agents' preferences, or they declare truthfully
- the goal: a social choice function: a mapping from everyone's
- preferences to a particular outcome, which is enforced
  - how to pick such functions with desirable properties?

Non-Ranking Voting Schemes

- Plurality
  - pick the outcome which is preferred by the most people
- Cumulative voting
  - distribute e.g., 5 votes each
  - possible to vote for the same outcome multiple times
- Approval voting
  - accept as many outcomes as you “like"

Ranking Voting Schemes

- Plurality with elimination (instant runoff)
  - everyone selects their favorite outcome
  - the outcome with the fewest votes is eliminated
  - repeat until one outcome remains
- Borda
  - assign each outcome a number.
  - The most preferred outcome gets a score of n-1, the next most preferred gets n-2, down to the nth outcome which gets 0.
  - Then sum the numbers for each outcome, and choose the one that has the highest score
- Pairwise elimination
  - in advance, decide a schedule for the order in which pairs will be compared.
  - given two outcomes, have everyone determine the one that they prefer
  - eliminate the outcome that was not preferred, and continue with the schedule

Condorcet Condition

- If there is a candidate who is preferred to every other candidate in pairwise runoff, that candidate should be the winner
- While the Condorcet condition is considered an important property for a voting system to satisfy, there is not always a Condorcet winner
- sometimes, there's a cycle where A defeats B, B defeats C, and C defeats A in their pairwise runoffs

Condorcet Example

499 agents: A > B > C

3 agents: B > C > A

498 agents: C > B > A

- What is the Condorcet winner?
- What would win under plurality voting?
- What would win under plurality with elimination?

Condorcet Example

499 agents: A > B > C

3 agents: B > C > A

498 agents: C > B > A

- What is the Condorcet winner? B
- What would win under plurality voting? A
- What would win under plurality with elimination? C

Sensitivity to Losing Candidate

35 agents: A > C > B

33 agents: B > A > C

32 agents: C > B > A

- What candidate wins under plurality voting?
- What candidate wins under Borda voting?
- Now consider dropping C. Now what happens under both Borda and plurality?

Sensitivity to Losing Candidate

35 agents: A > C > B

33 agents: B > A > C

32 agents: C > B > A

- What candidate wins under plurality voting? A
- What candidate wins under Borda voting? A
  A = 35*2+33 B =33*2+32 C = 32*2+35
- Now consider dropping C. Now what happens under both Borda and plurality? B wins.
  A = 35 B =33+32

Sensitivity to Agenda Setter

35 agents: A > C > B

33 agents: B > A > C

32 agents: C > B > A

- Who wins pairwise elimination, with the ordering A;B;C?
- Who wins with the ordering A;C;B?
- Who wins with the ordering B;C;A?

Sensitivity to Agenda Setter

35 agents: A > C > B

33 agents: B > A > C

32 agents: C > B > A

- Who wins pairwise elimination, with the ordering A;B;C? C
- Who wins with the ordering A;C;B? B
- Who wins with the ordering B;C;A? A

Another Pairwise Elimination Problem

- 1 agent: B D C A
- 1 agent: A B D C
- 1 agent: C A B D
- Who wins under pairwise elimination with the ordering A;B;C;D? D.
- What is the problem with this?
- all of the agents prefer B to D
- the selected candidate is Pareto-dominated!

THE MATHEMATICS OF DEMOCRACY

WHAT IS *Arrow's Impossibility Theorem?*

In order to find the "best" ranked voting system, we reviewed several criteria we want it to satisfy. **Arrow's Impossibility Theorem** says that this perfect voting system cannot exist: in an election with at least three candidates, if we want a voting system to satisfy both

- **Pareto Condition**:
  - if everyone prefers A to B, then B is not a winner; and
- **Independence of Irrelevant Alternatives (IIA)**:
  - a group's preference for one candidate over another should not depend on how the remaining candidates are arranged

...then that voting system must be a **dictatorship**, namely there will be a voter whose vote decides the election regardless of how everyone else voted.

Why can't any other voting system satisfy both Pareto criterion and IIA?

FIND THE ANSWER AT HTTPS://MATHEMATICS-DEMOCRACY-INSTITUTE.ORG/MATH-AND-POLITICS-TRIVIA/

- Social Choice
- Representative choice
- Gerrymandering

Representative choice

Party	Votes
A	1000
B	800
C	400

If we have 5 seats. How many seats per party?

https://en.wikipedia.org/wiki/Proportional_representation

Representative choice

Party	Votes
A	1000
B	800
C	400

If we have 5 seats. How many seats per party?

A	1000	45%	2	40%	3	60%
B	800	36%	2	40%	1	20%
C	400	18%	1	20%	1	20%
	2200		5		5	

https://en.wikipedia.org/wiki/Proportional_representation

D'Hondt method

- After all the, successive quotients are calculated for each party. The party with the largest quotient wins one seat, and its quotient is recalculated.

- Quot = V/(s+1)

- V is the total number of votes that party received, and
- s is the number of seats that party has been allocated so far, initially 0 for all parties.

The total votes cast for each party in the electoral district is divided, first by 1, then by 2, then 3, up to the total number of seats to be allocated for the district/constituency. Say there are p parties and s seats. Then a grid of numbers can be created, with p rows and s columns, where the entry in the ith row and jth column is the number of votes won by the ith party, divided by j. The s winning entries are the s highest numbers in the whole grid; each party is given as many seats as there are winning entries in its row.

Representative choice

	Votes	% Vote	FPTP Seats		D'Hondt Seats		Sainte-Laguë Seats		Hare-LR Seats	
Con	258,794	47.4%	8	72.7%	6	54.5%	6	54.5%	5	45.5%
Lab	204,011	37.4%	3	27.3%	5	45.5%	4	36.4%	4	36.4%
LD	33,604	6.2%					1	9.1%	1	9.1%
Brexit Party	15,728	2.9%							1	9.1%
Ashfield Ind	13,498	2.5%								
Green	10,375	1.9%								
Others	9,743	1.8%								
Deviation				25.3%		15.2%		10.1%		9.1%

https://electoral-reform.org.uk/what-is-the-difference-between-dhondt-sainte-lague-and-hare/

Gerrymandering

50 PRECINCTS
60% BLUE
40% RED

Who should be the winner?

If you can choose the districts…

Gerrymandering

https://magazine.caltech.edu/post/rethinking-redistricting

Multiagent decision making: Bayesian Games

AASMA 2023

Alberto Sardinha
jose.alberto.sardinha@tecnico.ulisboa.pt
Rui Prada
rui.prada@tecnico.ulisboa.pt

Outline

- **Introduction to Bayesian games**
- First definition
- Second definition
- Analyzing Bayesian games
- Exercises

Bayesian Games

- So far, all of the games make the following **assumption**:
  - **All agents know what game is being played**
- More specifically, we assume to be common knowledge for all agents:
  - the **number of agents**
  - the **actions** available to each agent
  - the **payoff** associated with each joint action

Bayesian Games

What if the agents are uncertain about the game being played?

Bayesian Games

Bayesian Games

Bayesian Games

Bayesian Games

- **Bayesian games** (or games of incomplete information) allow us to:
  - Represent **agents' uncertainties about the game being played**
  - The uncertainty regarding the game is represented as a **probability distribution** over a set of possible games

Bayesian Games

- Two key assumptions in Bayesian games:
  - The **games differ only in their payoffs**
    - Hence, all possible games have the same number of agents and the same action set for each agent
  - The **beliefs of the different agents are posteriors**
    - We obtain the **posteriors** by conditioning a **common prior** on **individual private signals**

Outline

- Introduction to Bayesian games
- **First definition**
- Second definition
- Analyzing Bayesian games
- Exercises

Bayesian Games – First Definition

- A Bayesian game consists of:
  - A set of games that differ only in their payoffs
  - A common prior defined over the games
  - A partition structure over the games for each agent

Bayesian Games – First Definition

- **Definition** (Bayesian game: information sets): *A* **Bayesian game** *is a tuple* (N, G, P, I) *where:*
  - N *is a set of agents*
  - G *is a set of games with* N *agents each*
    - *such that, if* $g, g' \in G$ *then for each agent* $i \in N$ *the action sets in* g *is identical to the action sets in* g'
  - $P \in \Pi(G)$ *is a common prior over games*
    - *where* $\Pi(G)$ *is the set of all probability distributions over* G
  - $I = (I_1, \dots, I_N)$ *is a tuple of partitions of* G*, one for each agent*

Bayesian Games – First Definition

There are four possible games that might be played

	$I_{2,1}$	$I_{2,2}$
$I_{1,1}$	MP: $\begin{array}{cc} 2,0 & 0,2 \\ 0,2 & 2,0 \end{array}$ $p = 0.3$	PD: $\begin{array}{cc} 2,2 & 0,3 \\ 3,0 & 1,1 \end{array}$ $p = 0.1$
$I_{1,2}$	Coord: $\begin{array}{cc} 2,2 & 0,0 \\ 0,0 & 1,1 \end{array}$ $p = 0.2$	BoS: $\begin{array}{cc} 2,1 & 0,0 \\ 0,0 & 1,2 \end{array}$ $p = 0.4$

MP = Matching Pennies
Coord = Coordination game
PD = Prisoner's dilemma
BoS = Battle of Sexes

Bayesian Games – First Definition

All games have the same number of agents

	$I_{2,1}$	$I_{2,2}$
$I_{1,1}$	MP: $\begin{array}{cc} 2,0 & 0,2 \\ 0,2 & 2,0 \end{array}$ $p = 0.3$	PD: $\begin{array}{cc} 2,2 & 0,3 \\ 3,0 & 1,1 \end{array}$ $p = 0.1$
$I_{1,2}$	Coord: $\begin{array}{cc} 2,2 & 0,0 \\ 0,0 & 1,1 \end{array}$ $p = 0.2$	BoS: $\begin{array}{cc} 2,1 & 0,0 \\ 0,0 & 1,2 \end{array}$ $p = 0.4$

Bayesian Games – First Definition

A common prior over the games

	$I_{2,1}$	$I_{2,2}$
$I_{1,1}$	MP: $\begin{matrix} 2,0 & 0,2 \\ 0,2 & 2,0 \end{matrix}$ $p = 0.3$	PD: $\begin{matrix} 2,2 & 0,3 \\ 3,0 & 1,1 \end{matrix}$ $p = 0.1$
$I_{1,2}$	Coord: $\begin{matrix} 2,2 & 0,0 \\ 0,0 & 1,1 \end{matrix}$ $p = 0.2$	BoS: $\begin{matrix} 2,1 & 0,0 \\ 0,0 & 1,2 \end{matrix}$ $p = 0.4$

E.g., there is 0.4 chance that the Agents will play BoS

Bayesian Games – First Definition

The action sets are the same in all games

	$I_{2,1}$		$I_{2,2}$	
	MP		PD	
	L	R	L	R
$I_{1,1}$ U	$2,0$	$0,2$	$2,2$	$0,3$
$I_{1,1}$ D	$0,2$	$2,0$	$3,0$	$1,1$
	$p = 0.3$		$p = 0.1$	
	Coord		BoS	
	L	R	L	R
$I_{1,2}$ U	$2,2$	$0,0$	$2,1$	$0,0$
$I_{1,2}$ D	$0,0$	$1,1$	$0,0$	$1,2$
	$p = 0.2$		$p = 0.4$	

Bayesian Games – First Definition

Information sets for agent 1
(indistinguishable sets)

$I_{2,1}$ $I_{2,2}$

$I_{1,1}$

MP

$2,0$	$0,2$
$0,2$	$2,0$

$p = 0.3$

PD

$2,2$	$0,3$
$3,0$	$1,1$

$p = 0.1$

$I_{1,2}$

Coord

$2,2$	$0,0$
$0,0$	$1,1$

$p = 0.2$

BoS

$2,1$	$0,0$
$0,0$	$1,2$

$p = 0.4$

E.g., agent 1 is going to find out that he is in this equivalence class

(i.e., either playing Coord or BoS, but not MP or PD)

E.g., if nature decides that the game being played is BoS

Bayesian Games – First Definition

Information sets for agent 2
(indistinguishable sets)

E.g., agent 2 is going to find out that she is in this equivalence class

(i.e., either playing PD or BoS, but not MP or Coord)

	$I_{2,1}$	$I_{2,2}$
$I_{1,1}$	MP: $\begin{array}{cc} 2,0 & 0,2 \\ 0,2 & 2,0 \end{array}$ $p = 0.3$	PD: $\begin{array}{cc} 2,2 & 0,3 \\ 3,0 & 1,1 \end{array}$ $p = 0.1$
$I_{1,2}$	Coord: $\begin{array}{cc} 2,2 & 0,0 \\ 0,0 & 1,1 \end{array}$ $p = 0.2$	BoS: $\begin{array}{cc} 2,1 & 0,0 \\ 0,0 & 1,2 \end{array}$ $p = 0.4$

E.g., if nature decides that the game being played is BoS

Bayesian Games – First Definition

- So what does this mean?
  - When the agents are **deciding what action to take**:
    - They decide **without fully knowing the game** that will be played
    - They have **to reason about the other agents** without knowing what the other agents are going to think

Bayesian Games – First Definition

So what do the agents know?

- They know everything about this setting:
  - The games
  - The common prior
  - The equivalence classes of all agents

	$I_{2,1}$	$I_{2,2}$
$I_{1,1}$	MP $\begin{array}{\|c\|c\|}\hline 2,0 & 0,2 \\ \hline 0,2 & 2,0 \\ \hline\end{array}$ $p = 0.3$	PD $\begin{array}{\|c\|c\|}\hline 2,2 & 0,3 \\ \hline 3,0 & 1,1 \\ \hline\end{array}$ $p = 0.1$
$I_{1,2}$	Coord $\begin{array}{\|c\|c\|}\hline 2,2 & 0,0 \\ \hline 0,0 & 1,1 \\ \hline\end{array}$ $p = 0.2$	BoS $\begin{array}{\|c\|c\|}\hline 2,1 & 0,0 \\ \hline 0,0 & 1,2 \\ \hline\end{array}$ $p = 0.4$

Outline

- Introduction to Bayesian games
- First definition
- **Second definition**
- Analyzing Bayesian games
- Exercises

Bayesian Games – Second Definition

- An **alternative definition** of Bayesian games
  - It is mathematically equivalent to the first definition
- This definition has a **different presentation**
  - It is based on **types -** a way of defining uncertainty directly over a game's payoff function

Bayesian Games – Second Definition

- **Definition** (Bayesian game: types): *A* **Bayesian game** *is a tuple* (N, A, θ, p, u) *where:*
  - N *is a set of agents*
  - $A = A_1 \times \ldots \times A_n$, *where* A_i *is the set of actions available to agent* i
  - $\Theta = \Theta_1 \times \ldots \times \Theta_n$, *where* Θ_i *is the type space of agent* i
  - $p: \Theta \longmapsto [0,1]$ *is a common prior over types*
  - $u = (u_1, \ldots, u_n)$, *where* $u_i: A \times \Theta \longmapsto \mathbb{R}$ *is the utility function for agent* i

Bayesian Games – Second Definition

Agent 2's types

$I_{2,1}$ $\theta_{2,1}$ $I_{2,2}$ $\theta_{2,2}$

Agent 1's types

$\theta_{1,1}$ $I_{1,1}$

$\theta_{1,2}$ $I_{1,2}$

MP

	L	R
U	2, 0	0, 2
D	0, 2	2, 0

$p = 0.3$

PD

	2, 2	0, 3
	3, 0	1, 1

$p = 0.1$

Coord

2, 2	0, 0
0, 0	1, 1

$p = 0.2$

BoS

2, 1	0, 0
0, 0	1, 2

$p = 0.4$

a_1	a_2	θ_1	θ_2	u_1	u_2
U	L	$\theta_{1,1}$	$\theta_{2,1}$	2	0
U	L	$\theta_{1,1}$	$\theta_{2,2}$	2	2
U	L	$\theta_{1,2}$	$\theta_{2,1}$	2	2
U	L	$\theta_{1,2}$	$\theta_{2,2}$	2	1
U	R	$\theta_{1,1}$	$\theta_{2,1}$	0	2
U	R	$\theta_{1,1}$	$\theta_{2,2}$	0	3
U	R	$\theta_{1,2}$	$\theta_{2,1}$	0	0
U	R	$\theta_{1,2}$	$\theta_{2,2}$	0	0

a_1	a_2	θ_1	θ_2	u_1	u_2
D	L	$\theta_{1,1}$	$\theta_{2,1}$	0	2
D	L	$\theta_{1,1}$	$\theta_{2,2}$	3	0
D	L	$\theta_{1,2}$	$\theta_{2,1}$	0	0
D	L	$\theta_{1,2}$	$\theta_{2,2}$	0	0
D	R	$\theta_{1,1}$	$\theta_{2,1}$	2	0
D	R	$\theta_{1,1}$	$\theta_{2,2}$	1	1
D	R	$\theta_{1,2}$	$\theta_{2,1}$	1	1
D	R	$\theta_{1,2}$	$\theta_{2,2}$	1	2

Outline

- Introduction to Bayesian games
- First definition
- Second definition
- **Analyzing Bayesian games**
- Exercises

Analyzing Bayesian Games

- We will reason about a Bayesian game using types (i.e., second definition)

- How?

  - **Bayesian Nash Equilibrium =** A plan of action for each player as a function of types that maximize each type's utility:

    - Expecting over the actions of the other players

    - Expecting over the types of the other players

Analyzing Bayesian Games

- **Given a Bayesian game** (N, A, Θ, p, u) with a finite set of agents, actions, and types

- We can **define strategies** as follows:

  - **Pure strategy**: $\alpha_i : \Theta_i \longmapsto A_i$
    - A mapping from **every type** (from agent i) **to an action** (from agent i)

Analyzing Bayesian Games

- Three standard notions of expected utility (depending on the timing of the decision)
  - ***ex-ante***
    - The agent knows nothing about anyone's actual type (including his own)
  - ***interim***
    - An agent knows her own type but not the types of the other agents
  - ***ex-post***
    - The agent knows all agents' types

Analyzing Bayesian Games

- **Given a Bayesian game** (N, A, θ, p, u) with a finite set of agents, actions, and types.

- Agent i's ***interim expected utility*** with respect to θ_i and a pure strategy profile α is:

$$EU_i(\alpha|\theta_i) = \sum_{\theta_{-i} \in \Theta_{-i}} p(\theta_{-i}|\theta_i)\, u_i(\alpha, \theta_i, \theta_{-i})$$

Analyzing Bayesian Games

- **Definition**: *A* Bayesian Nash equilibrium *is a pure strategy profile* a *that satisfies*

$$\alpha_i \in \underset{\alpha'_i}{\operatorname{argmax}}\, EU_i(\alpha'_i, \alpha_{-i} | \theta_i)$$

Best responses

for each i *and* $\theta_i \in \Theta_i$

- The above is defined based on the ***interim*** stage

Outline

- Introduction to Bayesian games
- First definition
- Second definition
- Analyzing Bayesian games
- **Exercises**

Exercise 1

- A sheriff faces an armed suspect and they must (simultaneously) decide whether to shoot the other or not

Exercise 1

- A sheriff faces an armed suspect and they must (simultaneously) decide whether to shoot the other or not, and:
  - The suspect is either a criminal with probability p or innocent with probability $1 - p$
  - The sheriff would rather shoot if the suspect shoots, but not if the suspect does not
  - The criminal would rather shoot even if the sheriff does not, as the criminal would be caught if it does not shoot
  - The innocent suspect would rather not shoot even if the sheriff shoots

Exercise 2

- Consider a game where a Firm is recruiting a Worker

Exercise 2

- A **worker** can be of **high ability**, in which case he would like to **work when he is hired**
- Or a **worker** can be of **low ability**, in which case he would rather **shirk** (i.e., to avoid work)
- The firm would want to hire the worker that will work but not the worker that will shirk
- The worker knows his ability level
- The firm does not know whether the worker is of high ability or low ability

Exercise 2

- The firm believes that the worker is of high ability with probability p and low ability with probability $1 - p$

- Most importantly, the firm knows that the worker knows his own ability level

Exercise 2

- To model this situation, we let Nature choose between a worker with **high** ability and **low** ability, with probabilities $\boldsymbol{p}$ and $\mathbf{1} - \boldsymbol{p}$, respectively.

- We then let the worker observe the choice of Nature, but we do not let the firm observe Nature's choice.

Exercise 2

- **Given a Bayesian game** (N, A, Θ, p, u) with a finite set of agents, actions, and types

- One can write the game in this exercise as a **Bayesian game** as follows:

  - $N = \{F, W\}$
  - $A_F = \{hire, dont\}, A_W = \{work, shirk\}$
  - $\Theta_F = \{t_f\}, \Theta_W = \{high, low\}$
  - $p(t_f, high) = p$
  - $p(t_f, low) = 1 - p$

F = firm
W = worker

Exercise 2

- One can write the game in this exercise as a **Bayesian game** as follows:

  - The utility function $u(a_F, a_W, \theta_F, \theta_W)$ is defined by the following tables:

$$u(a_F, a_W, \theta_F = t_f, \theta_W = high)$$

		worker	
		work	***shirk***
firm	***hire***	1, 2	0, 1
	dont	0, 0	0, 0

$$u(a_F, a_W, \theta_F = t_f, \theta_W = low)$$

		worker	
		work	***shirk***
firm	***hire***	1, 1	-1, 2
	dont	0, 0	0, 0

Exercise 2

- Consider:
  - $p = \frac{3}{4}$
  - the pure strategy profile $\alpha^* = (\alpha_F^*, \alpha_W^*)$ where
    - $\alpha_F^*(t_f) = hire$
    - $\alpha_W^*(high) = work$
    - $\alpha_W^*(low) = shirk$

  - Check if this pure strategy profile is a Bayes Nash equilibrium.

Thank You

jose.alberto.sardinha@tecnico.ulisboa.pt
rui.prada@tecnico.ulisboa.pt