Multi-agent Reinforcement Learning

Multi-agent Systems

Mirco Musolesi

mircomusolesi@acm.org

Definition of Multiagent Systems

- Several possible definitions:
  - *Multiagent systems are distributed systems of independent actors called agents that are each independently controlled but that interact with one another in the same environment.* (see: Wooldridge, “Introduction to Multiagent Systems”, 2002 and Tulys and Stone, “Multiagent Learning Paradigms”, 2018).
  - *Multiagent systems are systems that include multiple autonomous entities with (possibly) diverging information* (see Shoham and Leyton-Brown, “Multiagent systems”, 2009).

Definition of Multiagent Learning

- We will use the following definition of multiagent learning:
  - *"The study of multiagent systems in which one or more of the autonomous entities improves automatically through experience*".

Characteristics of Multiagent Learning

- Different scale:
  - A city or an ant colony or a football team.
- Different degree of complexity:
  - A human, a machine, a mammal or an insect.
- Different types of interaction:
  - Frequent interactions (or not), interactions with a limited number of individuals, etc.

Presence of Regularity

- It is fundamental that there is a certain degree of regularity in the system otherwise prediction of behaviour is not possible.

- Assumption: past experience is somehow predictive of future expectations.

- Dealing with non-stationarity is a key problem.

  - It is the usual problem of reinforcement learning at the end.

Potential Paradigms

- We will consider 5 paradigms:
  - Online Multi-agent Reinforcement Learning towards individual utility
  - Online Multi-agent Reinforcement Learning towards social welfare
  - Co-evolutionary learning
  - Swarm intelligence
  - Adaptive mechanism design

Online Reinforcement Learning towards Individual Utility

- One of the most-studied scenarios in multiagent learning is that in which multiple independent agents take actions in the same environment and learn online to maximise their own individual utility functions (i.e., expected returns).

- From a formal point view (game-theory point of view), this can be considered a repeated normal form game.

  - A *repeated game* is a game that is based of a certain number of repetitions.

  - *Normal form games* are games that are presented using a matrix.

    - As aside, an *extensive form game* is a game for which an explicit representation of the sequence of the players' possible moves, their choices at every decision point and the information about other player's move and relative payoffs are known.

Example: Prisoner's Dilemma

- The Prisoner's Dilemma is a classic 2-player game.
- Description of the "game": two prisoners committed a crime together and are being interrogated separately.
- If neither of them confesses to the crime (they both "cooperate"), then they will both get a small punishment (corresponding to a payoff of 5).
- If one of them confesses (or "defects"), but the other does not, then the one that confesses gets off for free (payoff of 10), but the other gets the worst punishment possible (payoff of 0).
- If they both defect, they get a worst punishment (payoff of 1)

Prisoners' Dilemma

	Defect	Cooperate
Defect	(1, 1)	(10, 0)
Cooperate	(0, 10)	(5, 5)

Example: Prisoner's Dilemma

- Normal games were initially introduced as one-shot game.
- The players know each other's full utility (reward) functions and play the game only once.
- In this setting, the concept of Nash equilibrium was introduced: a set of actions such that no player would be better off deviating given that the other player's actions are fixed.
- Games can have one or multiple Nash equilibria.
- In the Prisoner's Dilemma, the only Nash Equilibrium is for both agents to defect.

[13] Whitehead, J. H. C., "Simple Homotopy Types." If $W = 1$, Theorem 5 follows from (17:3) on p. 155 of S. Lefschetz, *Algebraic Topology*, (New York, 1942) and arguments in §6 of J. H. C. Whitehead, "On Simply Connected 4-Dimensional Polyhedra" (*Comm. Math. Helv.*, 22, 48–92 (1949)). However this proof cannot be generalized to the case $W \neq 1$.

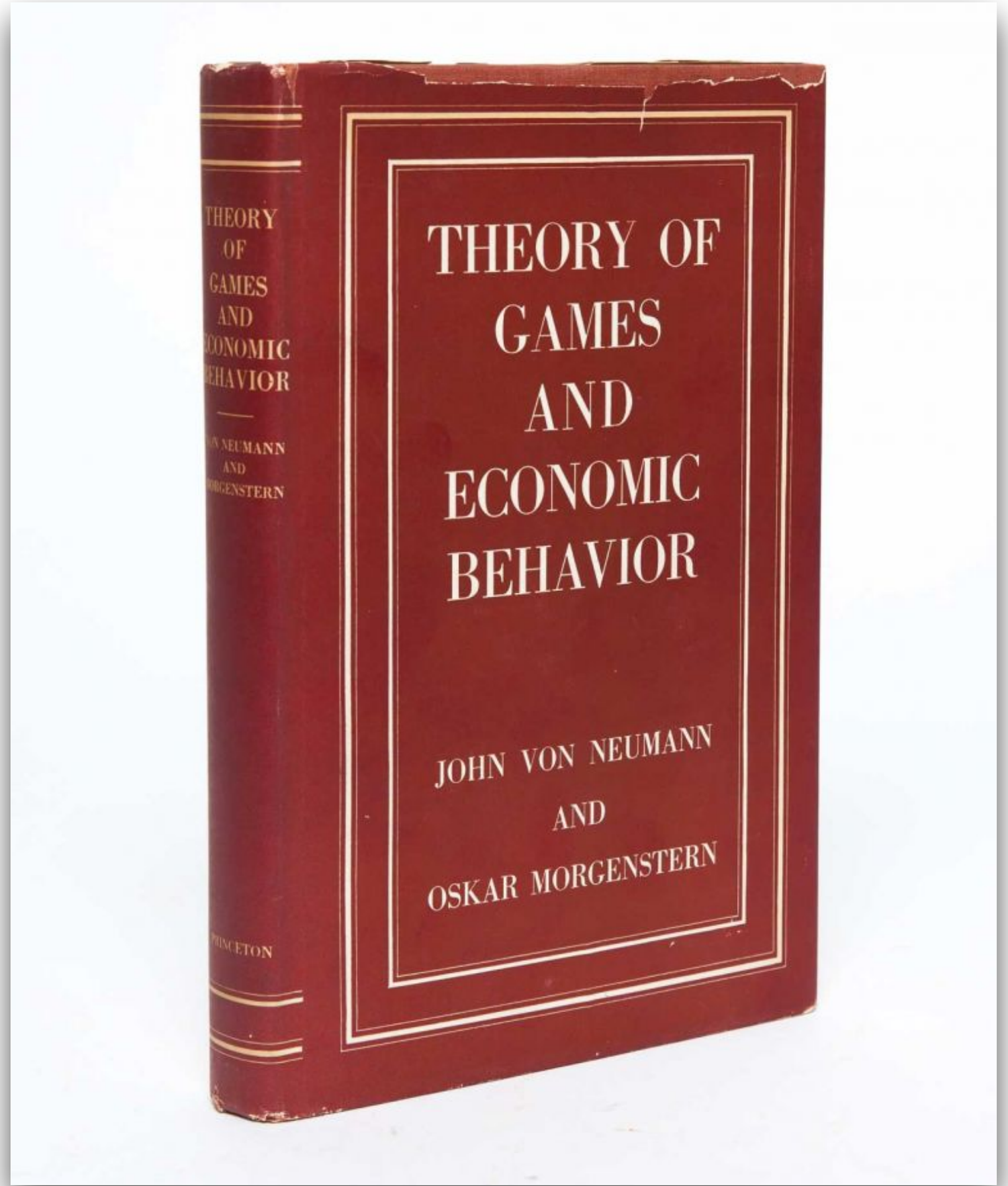
EQUILIBRIUM POINTS IN N-PERSON GAMES

By John F. Nash, Jr.*

Princeton University

Communicated by S. Lefschetz, November 16, 1949

One may define a concept of an n-person game in which each player has a finite set of pure strategies and in which a definite set of payments to the n players corresponds to each n-tuple of pure strategies, one strategy being taken for each player. For mixed strategies, which are probability


THEORY OF GAMES AND ECONOMIC BEHAVIOR
JOHN VON NEUMANN
AND
OSKAR MORGENSTERN
PRINCETON

Repeated Normal Form Games

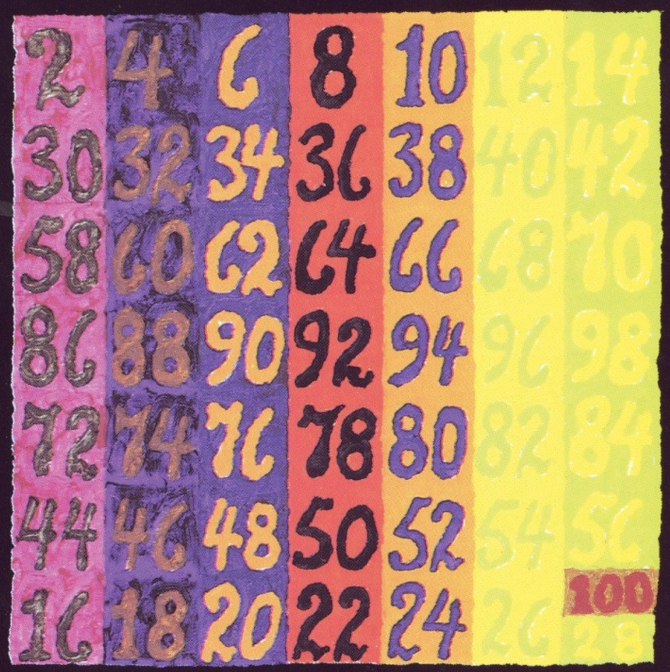
- In repeated normal form games, players interact with one another multiple times with the objectives of maximising their sum utilities (i.e., expected returns) over time.

- As you can imagine, Reinforcement Learning and possibly Deep Reinforcement Learning is well suited for this type of problems.

- Reinforcement Learning can also be used to understand the problem of the “evolution of cooperation” and the presence of altruism: why do we humans cooperate even if in presence of maximisation of personal reward function?

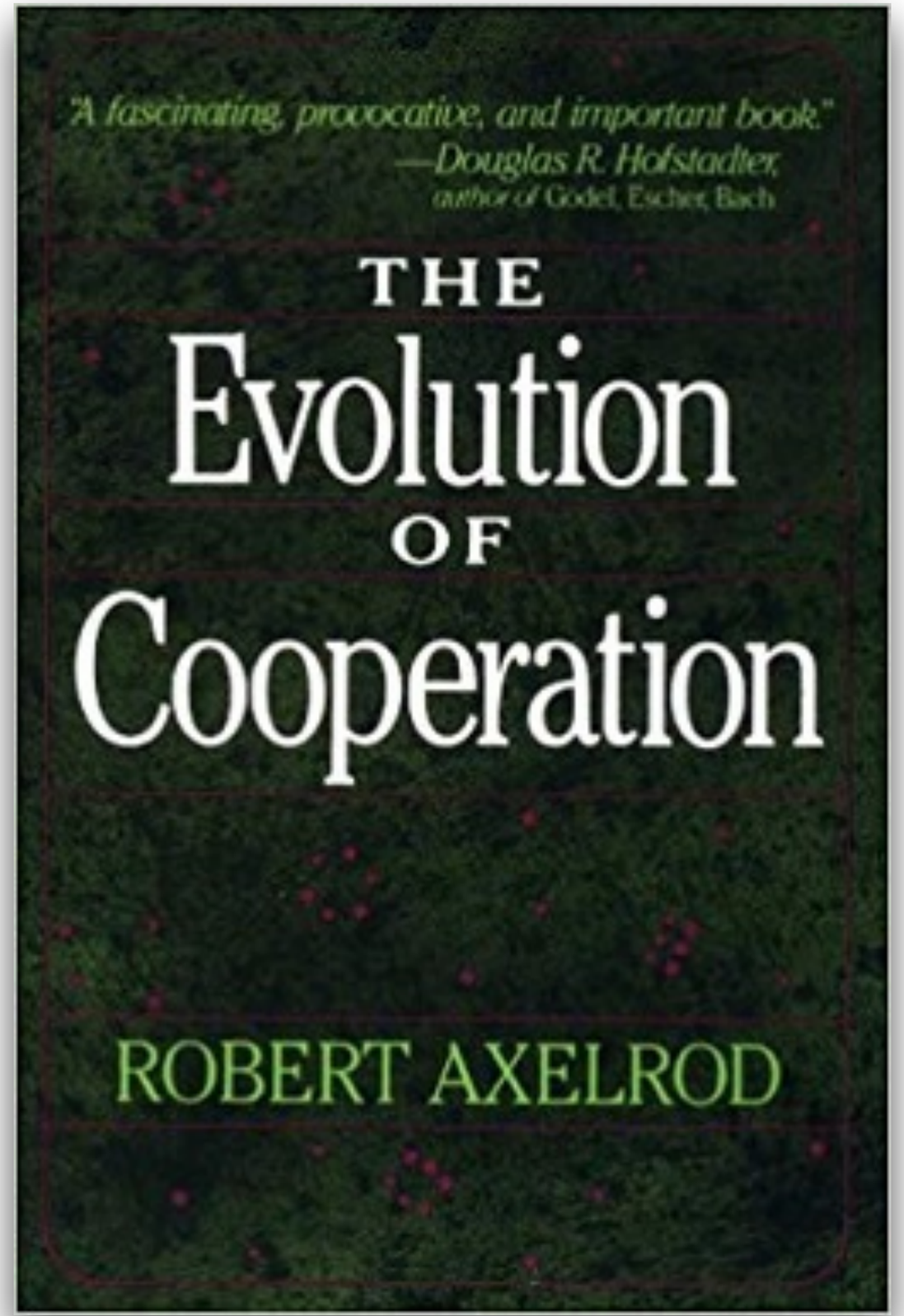
PRISONER'S DILEMMA

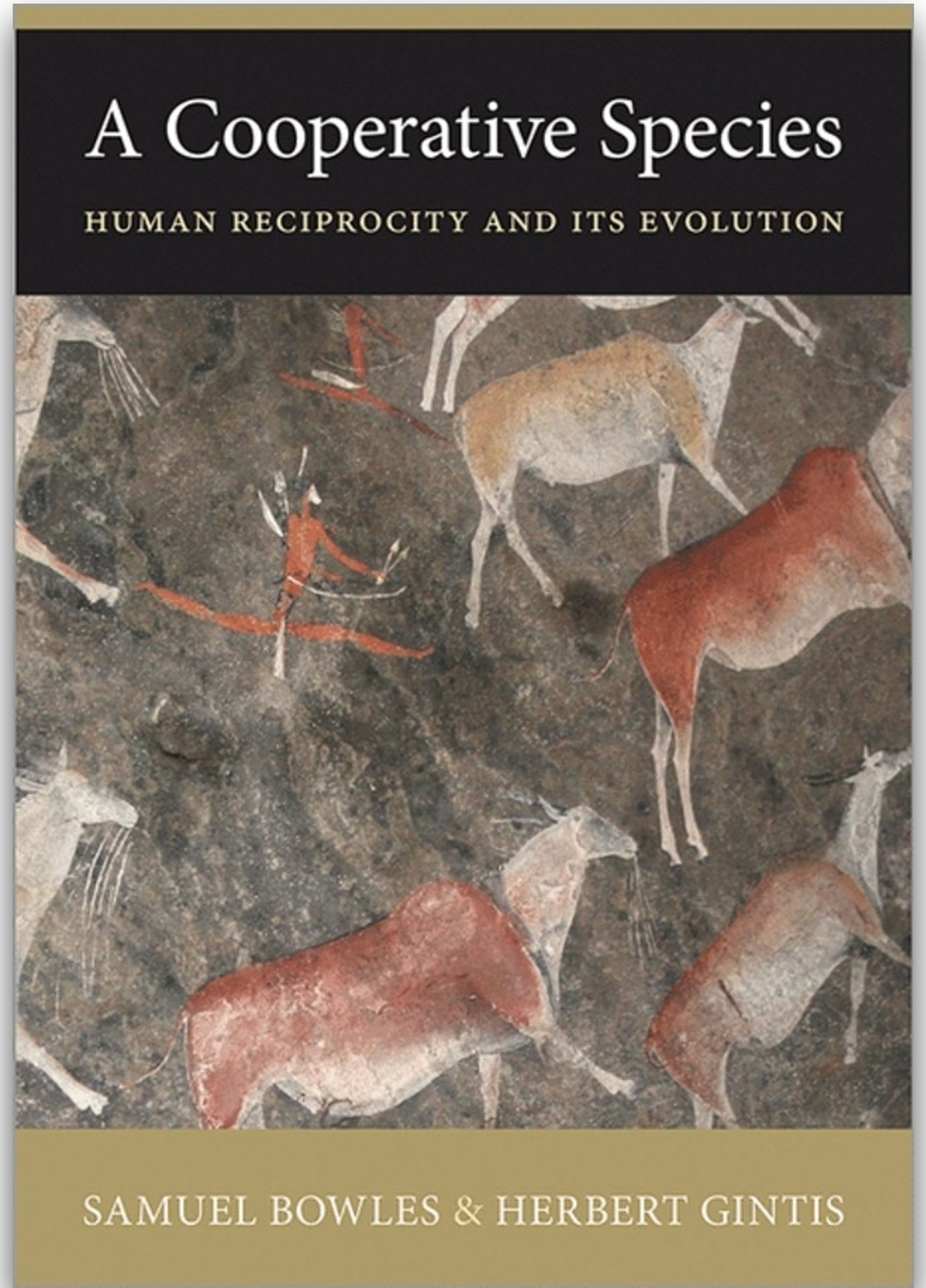
"Both a fascinating biography of von Neumann... and a brilliant social history of game theory and its role in the Cold War and nuclear arms race."
—San Francisco Chronicle



JOHN VON NEUMANN, GAME THEORY, AND THE PUZZLE OF THE BOMB

WILLIAM POUNDSTONE


"A fascinating, prooocative, and important book."
—Douglas R. Hofstadter,
author of Godel, Escher, Bach
THE
Evolution
OF
Cooperation
ROBERT AXELROD


A Cooperative Species
HUMAN RECIPROCITY AND ITS EVOLUTION
SAMUEL BOWLES & HERBERT GINTIS

Challenges in MARL: Credit Assignment

- One of the main practical problems in MARL is credit assignment, i.e., giving an actual credit (reward) for contributing to achieving the goal.
- This is particularly difficult in very complex games.
- Credit assignment is a very open problem in MARL in general.
- Solutions:
  - Equal reward/credit;
  - Heuristics for assignment;
  - Quantification of the contribution.

Predicting and Preventing Coordination Problems in Cooperative Q-learning Systems

Nancy Fulda and **Dan Ventura**
Computer Science Department
Brigham Young University
Provo, UT 84602
fulda@byu.edu, ventura@cs.byu.edu

Abstract

We present a conceptual framework for creating Q-learning-based algorithms that converge to optimal equilibria in cooperative multiagent settings. This framework includes a set of conditions that are sufficient to guarantee optimal system performance. We demonstrate the efficacy of the framework by using it to analyze several well-known multi-agent learning algorithms and conclude by employing it as a design tool to construct a simple, novel multi-agent learning algorithm.

ber of agents in the system increases. Some of them rely on global perceptions of other agents' actions or require a unique optimal equilibrium, conditions that do not always exist in real-world systems. As reinforcement learning and Q-learning are applied to real-world problems with real-world constraints, new algorithms will need to be designed.

The objective of this paper is to understand why the algorithms cited above are able to work effectively, and to use this understanding to facilitate the development of algorithms that improve on this success. We do this by isolating three factors that can cause a system to behave poorly: suboptimal individual convergence, action shadowing, and the equilibrium selection problem. We prove that the absence of these three

IJCAI 2007

Challenges in MARL: Non-stationarity

- Non-stationarity is a major problem in single agent RL: the problem is even more apparent in case of multi-agent RL systems.
- In fact, in multi-RL systems, the other agents are actually part of the environment.
  - In many situations, the state might also contain information about the actions of the other agents.
- If the behaviour of the other agents change, there might be an effect on the non-stationarity of the system under observation.
- What are the potential countermeasures?

Challenges in MARL: Training the Agents

- Training multi-agent systems is still an open problem, given the complexity of the problem in terms of state space (and potentially action space).

- A variety of the methods have been proposed with a major distinction between centralised vs decentralised methods.

- It is worth noting that execution can also be either centralised or decentralised.

Centralised vs Decentralised Training

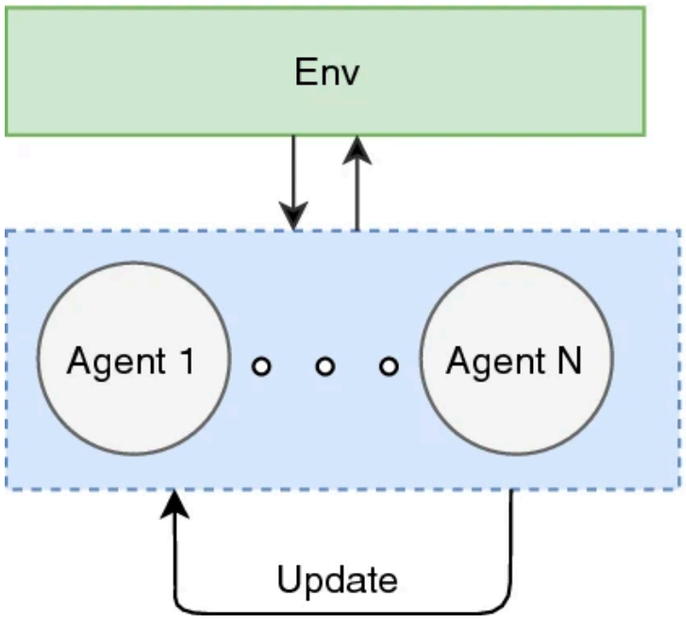
- In *centralised training*, policies are updated based on the mutual exchange of information during training.
  - This information is usually removed at deployment time.
- In *decentralised training*, each agent performs updates independently and develops an individual policy without utilising information from the other agents.

Centralised vs Decentralised Execution

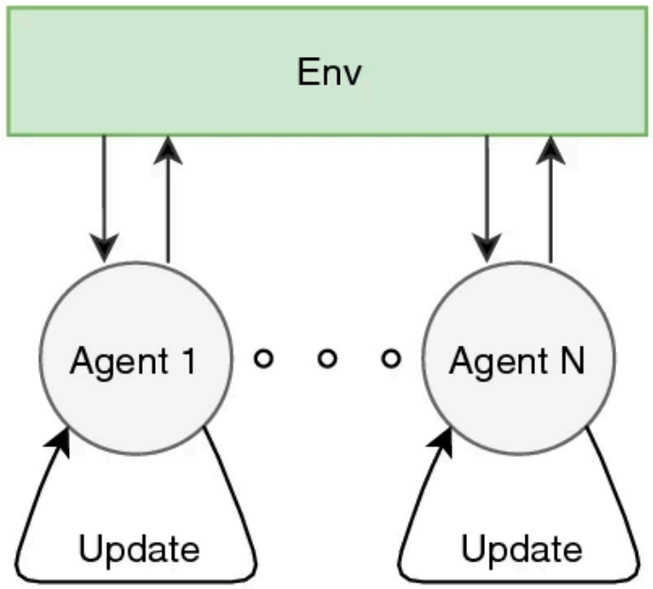
- In *centralised execution* agents are guided from a central unit (model), which computes the joint actions for all agents. In other words, the set of actions for all the agents are selected by this central unit.

- In *decentralised execution,* agents select the action to be executed independently.

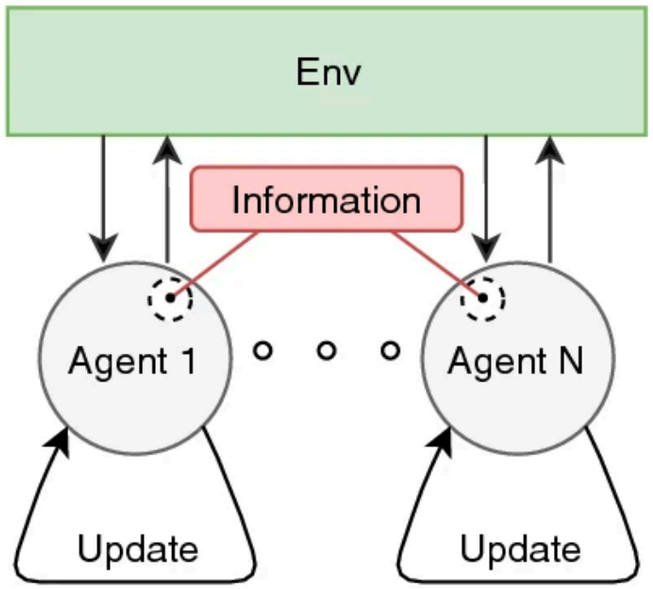
Training and Execution Models in MARL



Centralised
Training
Centralised
Execution



Decentralised
Training
Decentralised
Execution



Centralised
Training
Decentralised
Execution

Picture from Sven Gronauer and Klaus Diepold. Multi-agent deep reinforcement learning: a survey. Artificial Intelligence Review. 5. 2022

Online Reinforcement Learning towards Social Welfare

- We will now consider a practical case in which agents achieve cooperation through decentralised training and execution.

- In this scenario, multiple independent agents take actions in the same environment and learn online to maximise a global utility function.

- These are also called coordination games, where different players coordinate to achieve a given objective (i.e., global expected return).

- We will know consider a specific example using the Atari testbed for evaluation.

Multiagent Cooperation and Competition with Deep Reinforcement Learning

Ardi Tampuu* **Tambet Matiisen***

Dorian Kodelja **Ilya Kuzovkin** **Kristjan Korjus**

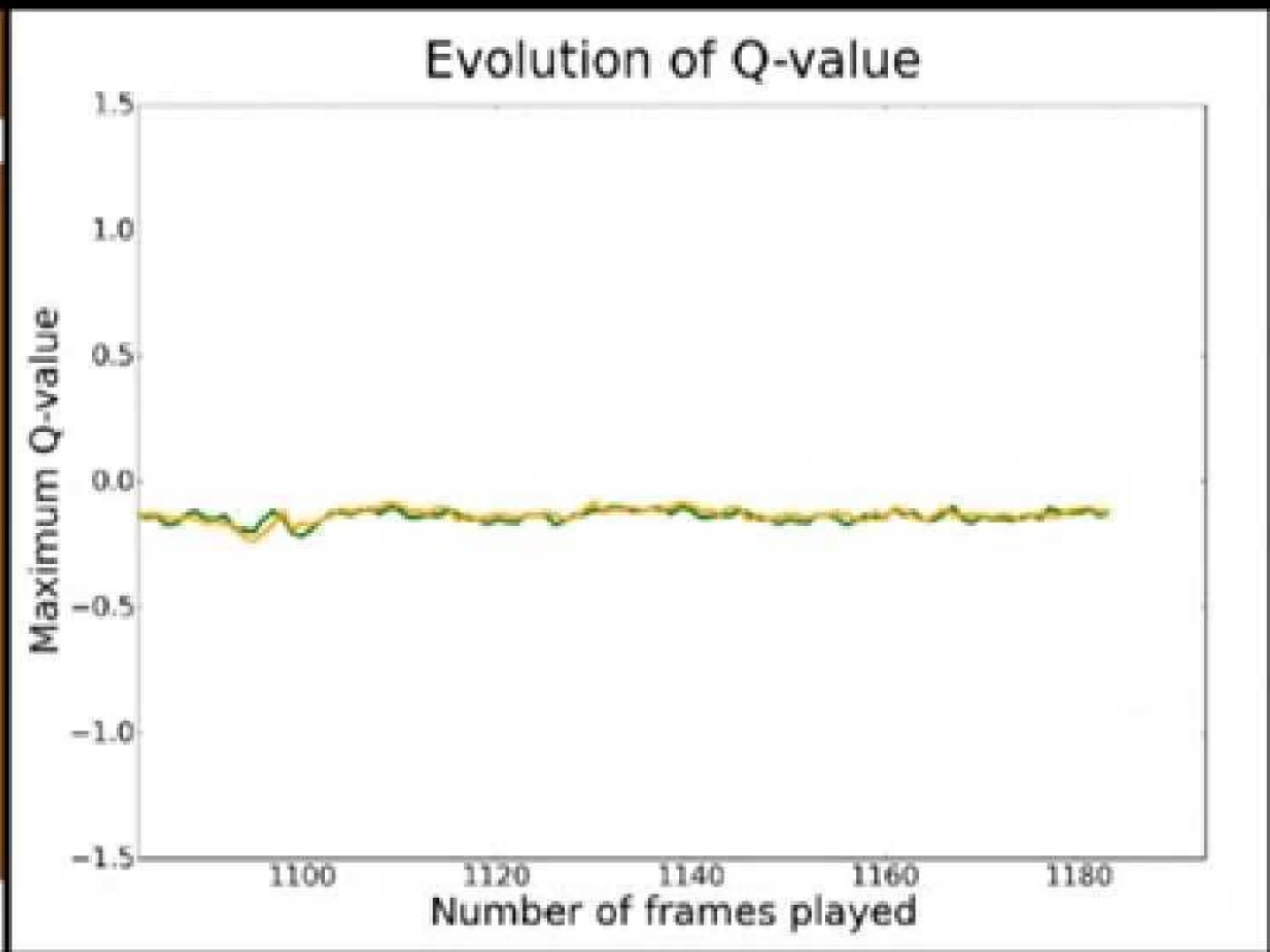
Juhan Aru[†] **Jaan Aru** **Raul Vicente**[✉]

Computational Neuroscience Lab, Institute of Computer Science, University of Tartu
[†] Department of Mathematics, ETH Zurich
ardi.tampuu@ut.ee, tambet.matiisen@ut.ee, raul.vicente.zafra@ut.ee

** these authors contributed equally to this work*

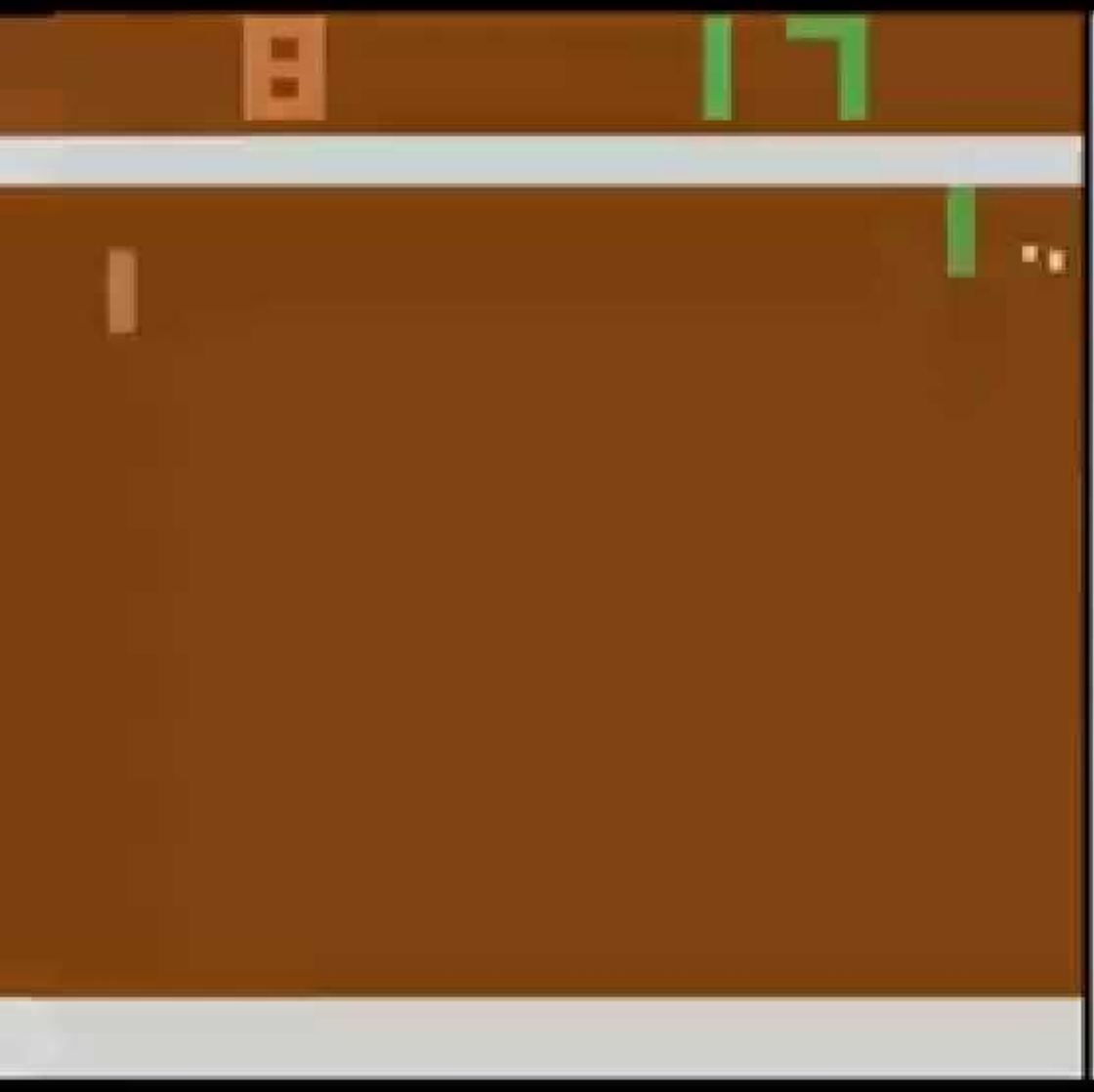
Abstract

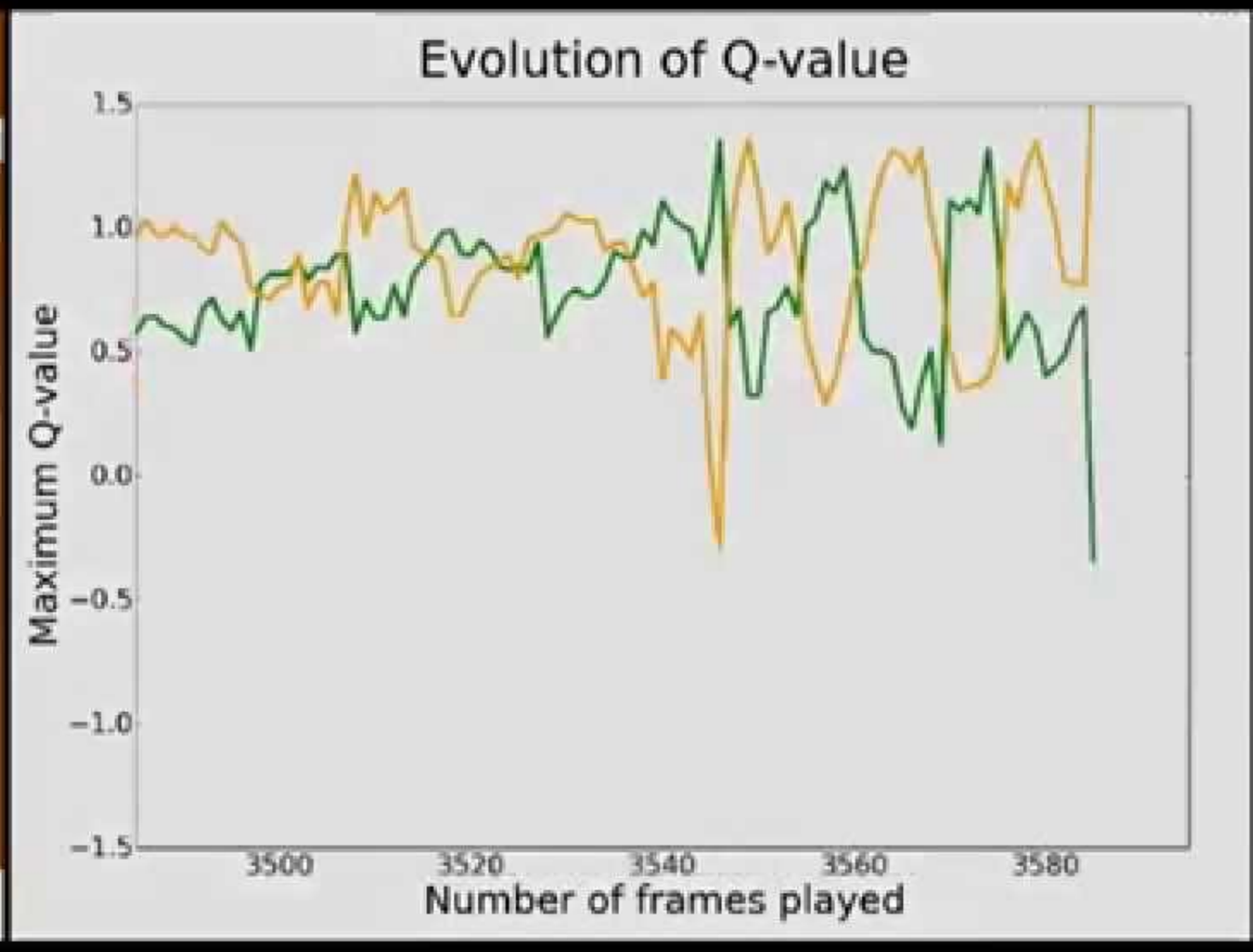
Multiagent systems appear in most social, economical, and political situations. In the present work we extend the Deep Q-Learning Network architecture proposed by Google DeepMind to multiagent environments and investigate how two agents controlled by independent Deep Q-Networks interact in the classic videogame *Pong*. By manipulating the classical rewarding scheme of Pong we demonstrate how competitive and collaborative behaviors emerge. Competitive agents learn to play and score efficiently. Agents trained under collaborative rewarding schemes find an optimal strategy to keep the ball in the game as long as possible. We also describe the progression from competitive to collaborative behavior. The present work demonstrates that Deep Q-Networks can become a practical tool for studying the decentralized learning of multiagent systems living in highly complex environments.



... but are sometimes capable of reaching a state where they never loose the ball.

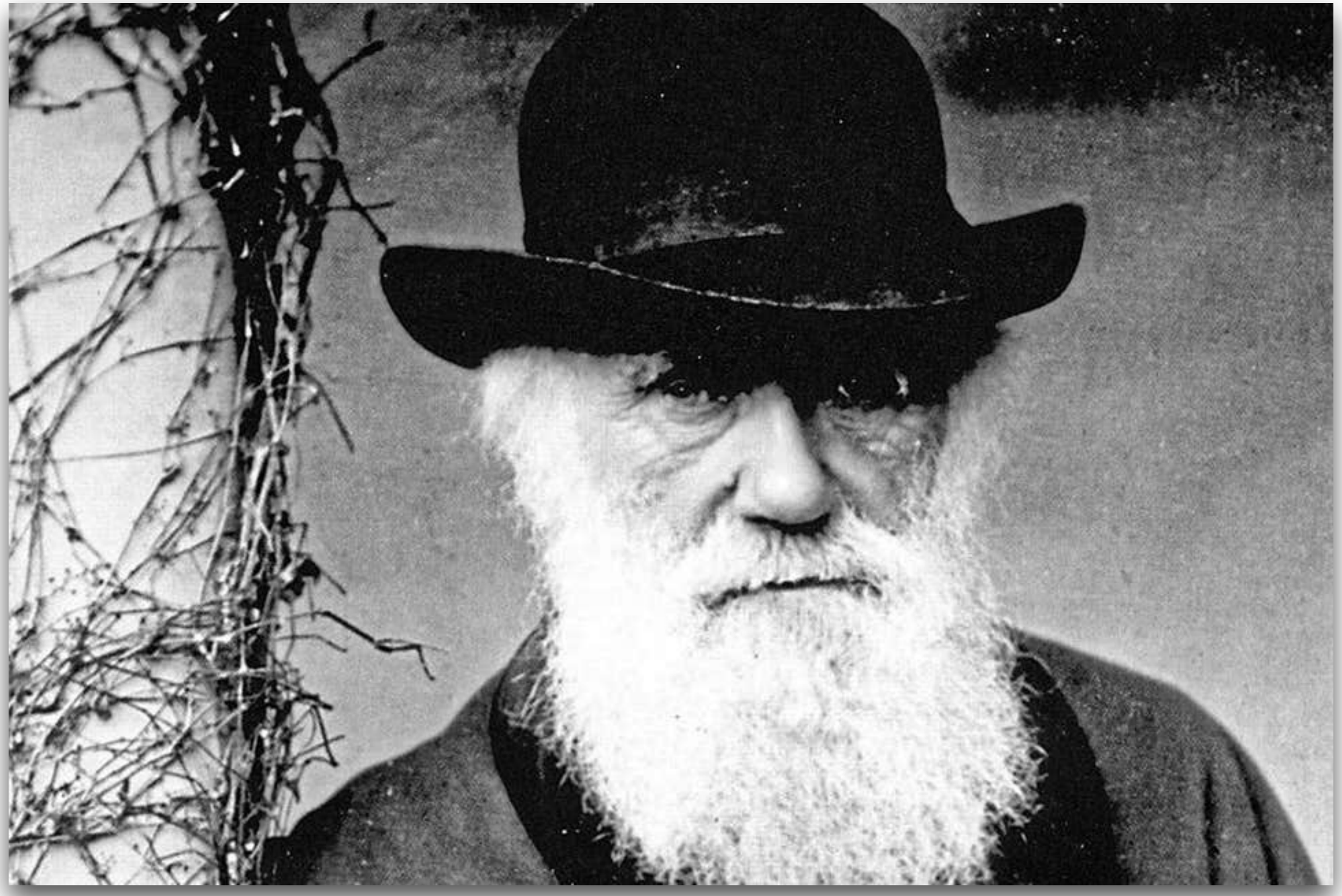
https://www.youtube.com/watch?v=Gb9DprIgdGw





The agents have learned to deal better with fast and bouncing balls.

https://www.youtube.com/watch?v=nn6_GUVDnVw



Co-evolutionary Approaches

- Evolution can be also used to model and learn agent behaviour as well. According to this paradigm, abstract Darwinian models of evolution are applied to to refine populations of agents (knows as *individuals*).

- These represent candidate solutions to a given problem.

- This process, usually called a *genetic algorithm*, consists of five steps: *representation*, *selection*, *generation of new individuals* (crossover and mutation), *evaluation* and *replacement*.

Evolutionary Algorithms

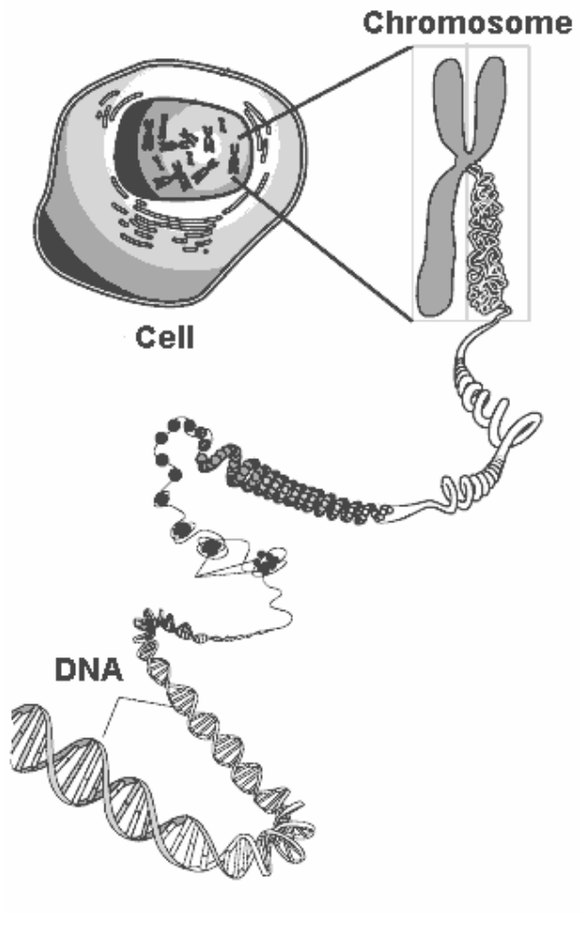
- An evolutionary algorithm begins with an initial population of randomly-generated agents. Each member of this population is then evaluated and assigned a fitness value.

- The evolutionary algorithm then uses a fitness oriented procedure to select agents, breeds and mutates them to produce child agents, which are then added to the population, replacing older agents.

- One evaluation, selection and breeding cycle is known as a *generation.*

- Successive generations refine a population.

  - You have a given set goal and you might have a time budget.

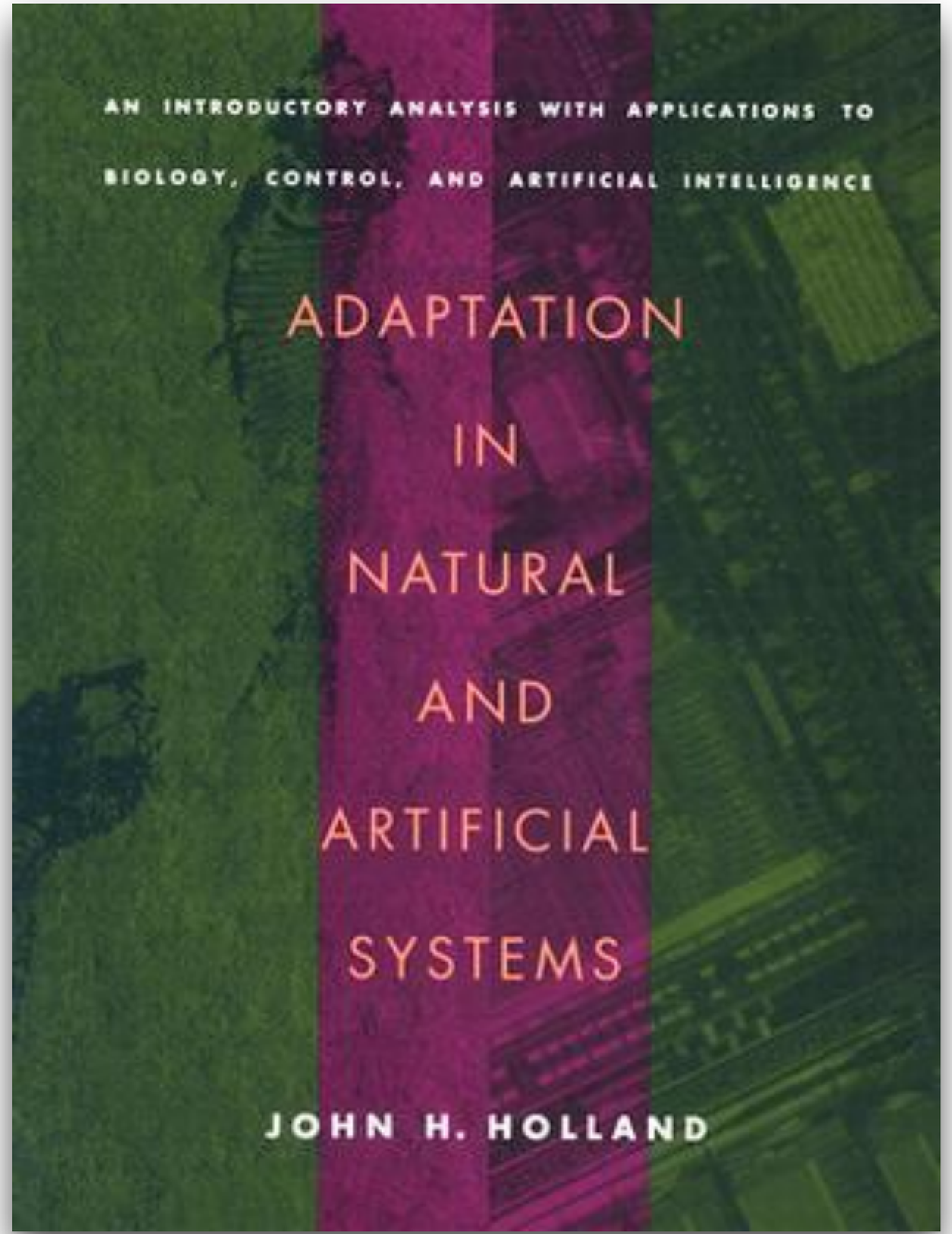


Selection Cycle

Images from Dario Floreano and Claudio Mattiussi. Bio-Inspired Artificial Intelligence. MIT Press 2011.

Problem of Representation

- Typically you encode a “vector” of information using binary coding (for example 4 bits for element of the vector).
  - This was the original version proposed by John Holland in 1970.
- Floating point representations are possible.
- Extensions include the use of real numbers.


AN INTRODUCTORY ANALYSIS WITH APPLICATIONS TO
BIOLOGY, CONTROL, AND ARTIFICIAL INTELLIGENCE
ADAPTATION
IN
NATURAL
AND
ARTIFICIAL
SYSTEMS
JOHN H. HOLLAND

Fitness Function

- You evaluate the performance of the phenotype (i.e., the actual performance of the behavior of your agent encoded through this genotype).

- There is a clear mapping with the biological analogy.

  - Darwin's "survival of the fittest".

- From a practical point of view, you can think about a performance measure as we did in deep learning.

Selection

- At each generation, only some of the individuals reproduce.
- The probability that an individual will make offsprings will be proportional to their fitness.
- One possibility is to have proportionate selection, i.e., the probability that an individual makes an offspring is proportional to how good its fitness is with respect to the population fitness.
- The probability of reproduction will be:

$$p_i = \frac{fitness_i}{\sum_j fitness_j}$$

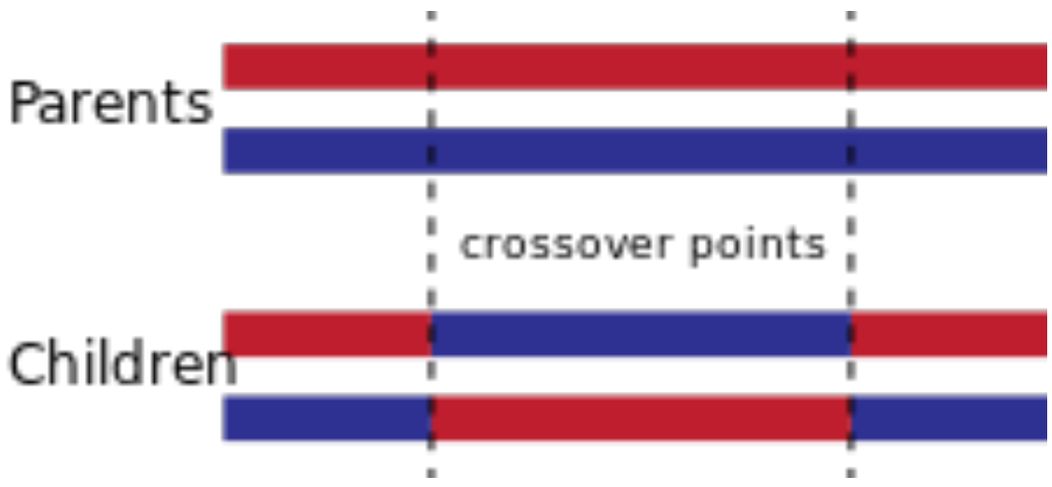
with the sum at the denominator over the entire population.

- You might have a system where only the top K individuals will reproduce (i.e., you sub-select first a set of individuals with high fitness and then you apply the form above).

Cross-over

- In genetic algorithms, crossover (recombination) is an operator that combine the genetic information of two parents to generate offsprings.

- It is one way to stochastically generate new solutions from an existing population and it is analogous to the crossover that happens during biological sexual reproduction.



Source: Wikimedia

Mutation

- Mutation is an operator used to maintain genetic diversity from one generation to the next (see biological mutation).

- Mutation alters one or more bits in the representation (chromosome).

- Example: bit string mutation (one or more bits)

  - 1010**1** -> 1010**0**

- A variety of mutation types have been explored (with different distribution, for groups of bits, flipping bits, etc.)

Evaluation and Replacement

- At each generation, the fitness of each individual is evaluated and using the mechanisms described above, all the entire population is usually entirely replaced by offspring (like in a real biological situation).

- Alternative solutions include an “elitist” solution where we maintain the n best individuals from the previous generation to prevent loss of the best individuals from the population (for example because of the effects of mutations or sub-optimal fitness evaluation).

Coevolution

- Coevoution is an extension of evolutionary algorithms for domains with multiple agents.

- Using evolutionary algorithms, we can train a policy to perform a state to action mapping. In this approach, rather than update the parameters of a single agent interacting with the environment as is done in reinforcement learning, one searches through a *population of policies* that have the highest fitness for the task at hand.

- For example we can use a probability vector as a representation of the policy.

- Alternatives include the use of evolutionary algorithms for estimating the hyper-parameters of the networks.

Swarm Intelligence

- Swarm intelligence is a bio-inspired machine learning technique based on the behaviour of social insects (e.g., ants and honeybees).

- The goal is to develop self-organised and decentralised adaptive algorithms.

- The learning is based on a large number of agents (usually with limited “computation” capabilities) that locally interact.

- The idea is to develop algorithms that lead to the emergence of cooperative behaviour in the population.

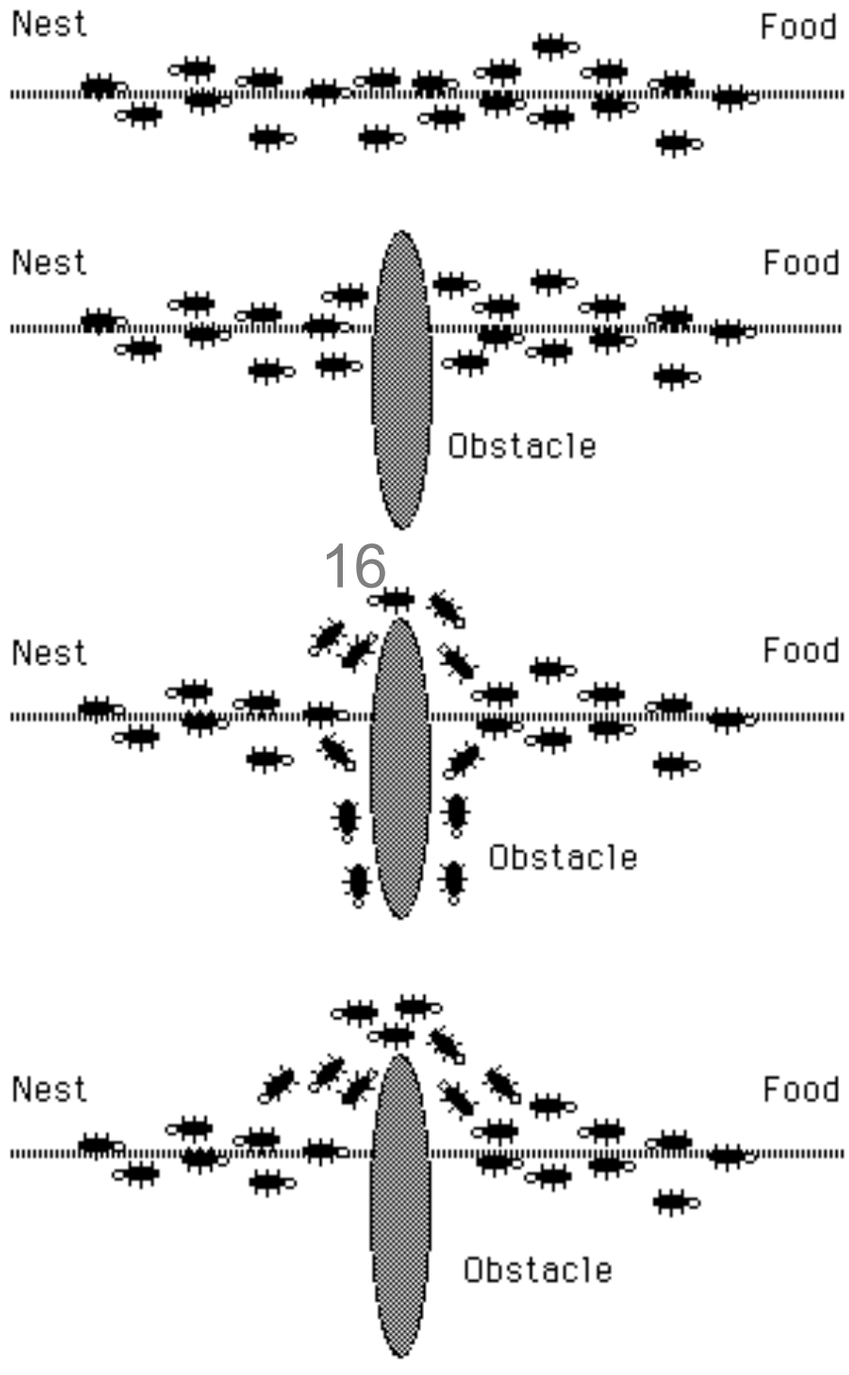
- Complex behaviour from simple local rules.

Swarm Intelligence

Video from Dario Floreano and Claudio Mattiussi. Bio-Inspired Artificial Intelligence. MIT Press 2011.

Stigmergy

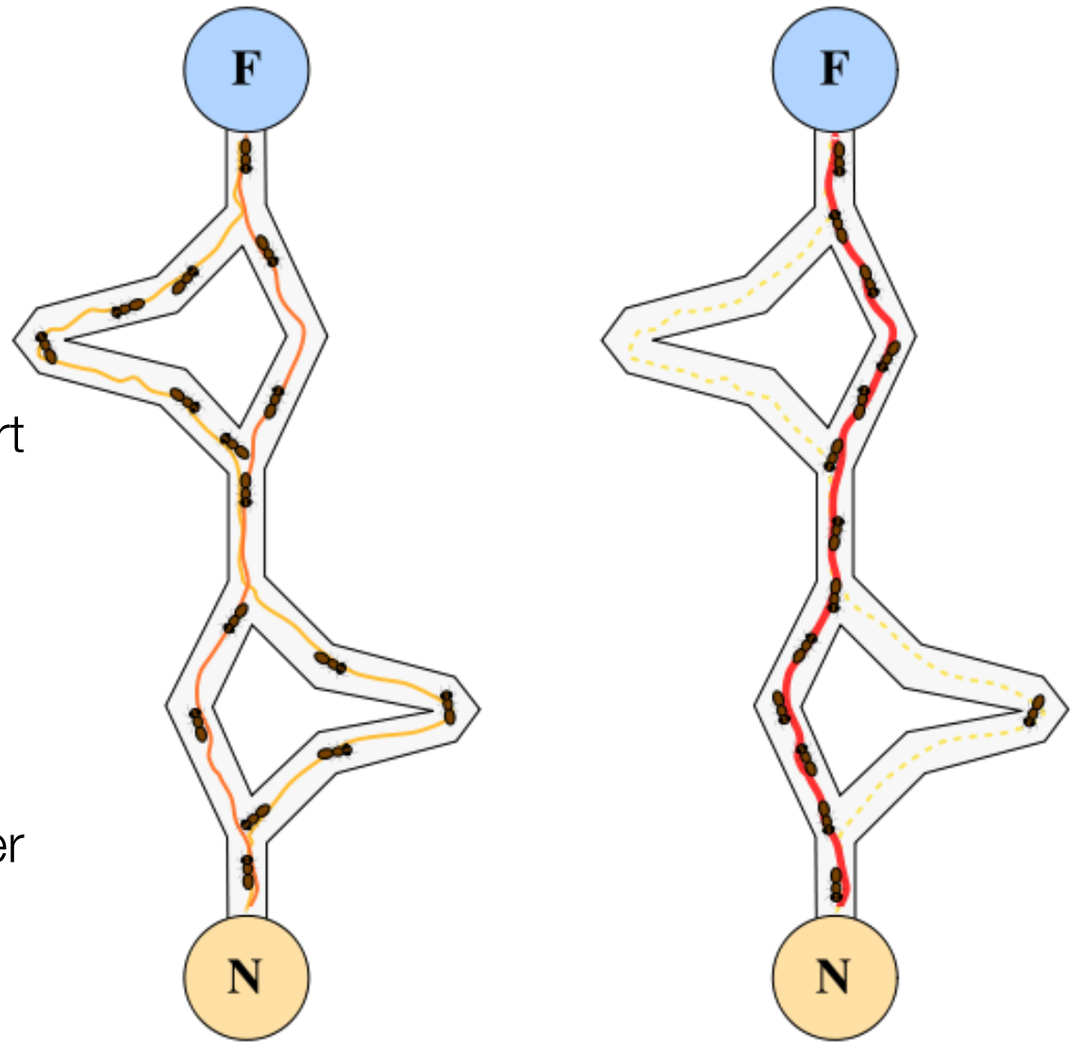
- The term *stimergy* indicates communication among individuals through modification of the environment.
- For example, some ants leave a chemical (pheromone) trail behind to trace the path.
- The chemical decays over time.
- This allows other ants to find the paths between the food and the nest.
- It also allows ants to find the *shortest path* among alternatives.



Source: Dario Floreano and Claudio Mattiussi. Bio-Inspired Artificial Intelligence. MIT Press 2011.

Stigmergy and Shortest Paths

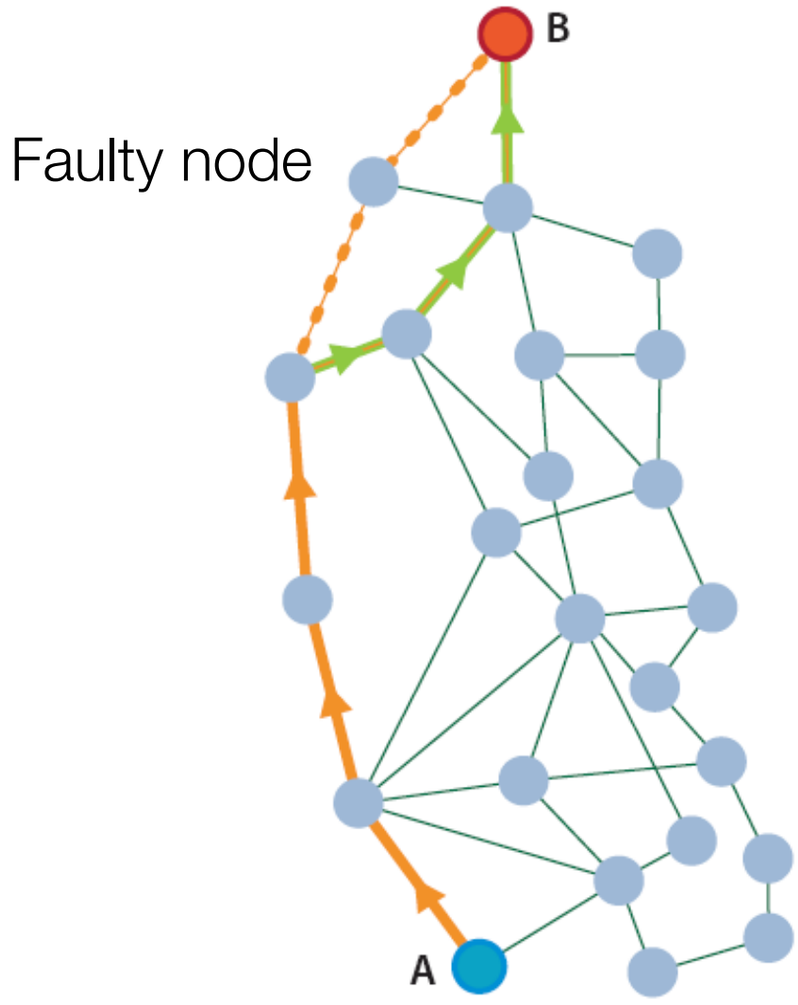
- As they move, ants deposit pheromone.
- Pheromone decays in time.
- Ants follow the paths with the highest pheromone concentration.
- Without pheromone, equal probability of choosing short or long path.
- Shorter paths allow for higher number of passages (it takes less time to go back and forth!)
- Therefore, pheromone level will be higher on the shorter path.
- Ants will increasingly tend to choose the shorter path.



Source: Dario Floreano and Claudio Mattiussi. Bio-Inspired Artificial Intelligence. MIT Press 2011.

Ant Colony Optimization

- Ant Colony Optimization is an algorithm developed by Dorigo et al. inspired upon stigmergic communication to find the shortest path in a network.



- Typical examples are Internet/computer networks problems and other problems that can be described by the Travel Salesman Problem.

- Other problems include scheduling of robots and coverage of areas (represented as networks).

Source: Dario Floreano and Claudio Mattiussi. Bio-Inspired Artificial Intelligence. MIT Press 2011.

Ant Colony Optimization: A New Meta-Heuristic

Marco Dorigo
IRIDIA
Université Libre de Bruxelles
mdorigo@ulb.ac.be

Gianni Di Caro
IRIDIA
Université Libre de Bruxelles
gdicaro@iridia.ulb.ac.be

Abstract- Recently, a number of algorithms inspired by the foraging behavior of ant colonies have been applied to the solution of difficult discrete optimization problems. In this paper we put these algorithms in a common framework by defining the Ant Colony Optimization (ACO) meta-heuristic. A couple of paradigmatic examples of applications of these novel meta-heuristic are given, as well as a brief overview of existing applications.

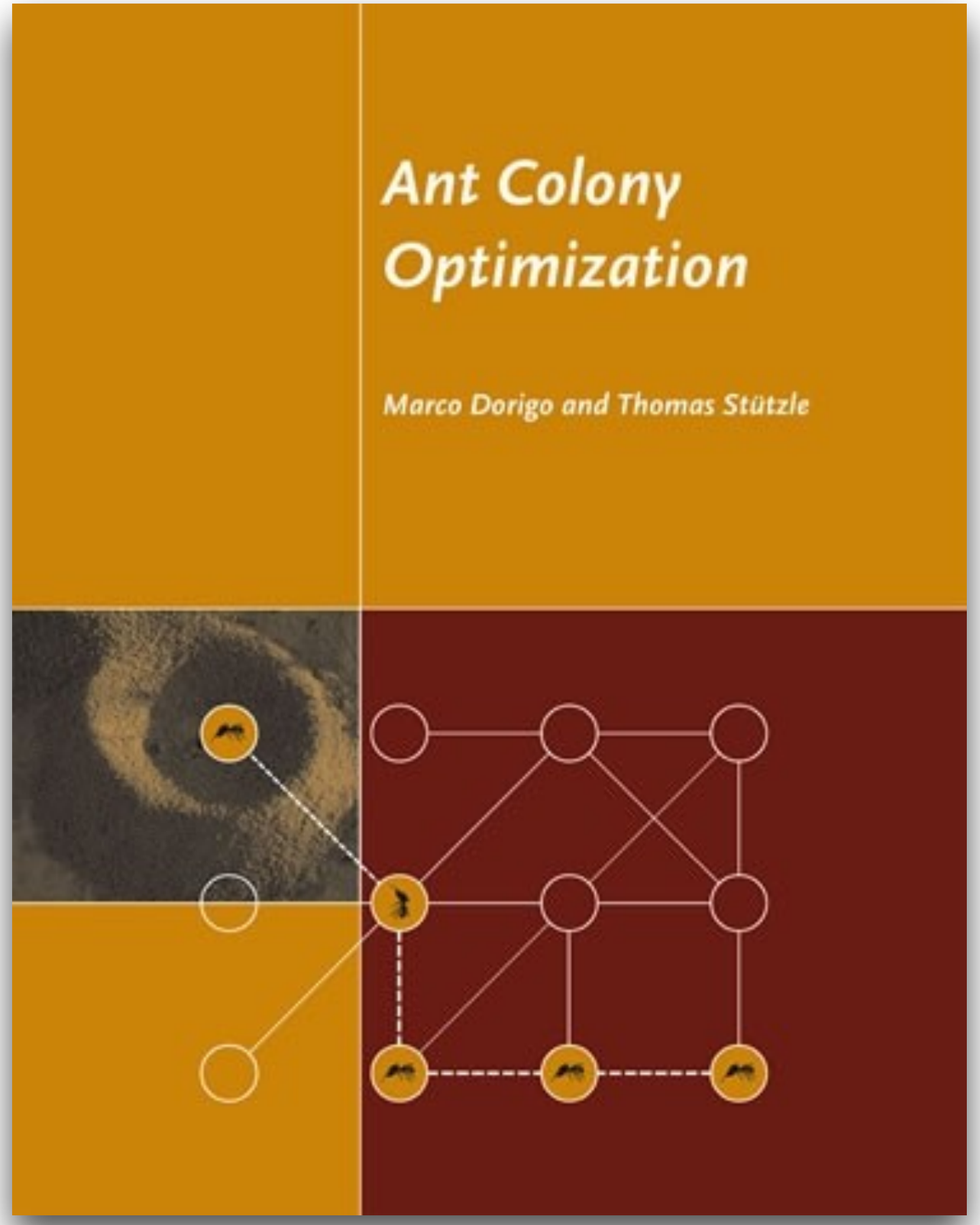
1 Introduction

In the early nineties an algorithm called *ant system* was proposed as a novel heuristic approach for the solution of combinatorial optimization problems (Dorigo *et al.*, 1991; Dorigo, 1992; Dorigo *et al.*, 1996). Ant system (AS), which was first applied to the traveling salesman problem, was recently extended and/or modified both to improve its performance and to apply it to other optimization problems. Improved versions of AS include, among others, ACS (Dorigo & Gambardella, 1997), $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System (Stützle & Hoos, 1998b), and AS_{rank} (Bullnheimer *et al.*, 1997b). All these algorithms have been applied to the TSP with varying degrees of success, but always improving over AS perfor-

2 The ACO meta-heuristic

The ACO meta-heuristic can be applied to discrete optimization problems characterized as follows.

- $C = \{c_1, c_2, \ldots, c_{N_C}\}$ is a finite set of *components*.
- $L = \{l_{c_i c_j} \mid (c_i, c_j) \in \tilde{C}\}$, $|L| \leq N_C^2$ is a finite set of possible *connections/transitions* among the elements of $\tilde{C}$, where $\tilde{C}$ is a subset of the Cartesian product $C \times C$.
- $J_{c_i c_j} \equiv J(l_{c_i c_j}, t)$ is a *connection cost* function associated to each $l_{c_i c_j} \in L$, possibly parameterized by some time measure t.
- $\Omega \equiv \Omega(C, L, t)$ is a finite set of *constraints* assigned over the elements of C and L.
- $s = \langle c_i, c_j, \ldots, c_k, \ldots \rangle$ is a sequence over the elements of C (or, equivalently, of L). A sequence s is also called a *state* of the problem. If S is the set of all possible sequences, the set $\tilde{S}$ of all the (sub)sequences that are feasible with respect to the constraints $\Omega(C, L, t)$, is a subset of S. The elements in $\tilde{S}$ define the problem's *feasible states*. The length of a sequence s, that is, the number of components in the sequence, is expressed by $|s|$.


Ant Colony
Optimization
Marco Dorigo and Thomas Stützle

Adaptive Mechanism Design

- It is also possible to think of a multi-agent learning setting in which the agents are fixed (or not controllable by us), but the interaction mechanism is to be learned.
- Typical example is an auction with a population of bidders.
  - Several parameters can be controlled:
    - Minimum price, simultaneous or non-simultaneous actions, mechanism (English auction, Vickrey auction, Dutch auction, etc.).
  - In this case, the auction house is not able to control the bidders (interacting agents), but the rules of interaction.
  - The parameters can be learned and refined over time (mechanism design).
- Other applications: frequency bidding, design of competition markets, etc.

Adaptive Mechanism Design

- In terms of adaptive mechanism design, there is also a strong interest from Economics.

- There is an emerging area of applications of Machine Learning and Reinforcement Learning to Economics.

- We have presented the different “categories” separately, but the “real world” is more complex thant hat.

The AI Economist: Improving Equality and Productivity with AI-Driven Tax Policies

Stephan Zheng[*,1], Alexander Trott[*,1], Sunil Srinivasa[1], Nikhil Naik[1], Melvin Gruesbeck[1], David C. Parkes[1,2], and Richard Socher[1]

[1]Salesforce Research
[2]Harvard University

April 28, 2020

Abstract

Tackling real-world socio-economic challenges requires designing and testing economic policies. However, this is hard in practice, due to a lack of appropriate (micro-level) economic data and limited opportunity to experiment. In this work, we train social planners that discover tax policies in dynamic economies that can effectively trade-off economic equality and productivity. We propose a two-level deep reinforcement learning approach to learn *dynamic tax policies*, based on economic simulations in which both agents and a government learn and adapt. Our data-driven approach does not make use of economic modeling assumptions, and learns from observational data alone. We make four main contributions. First, we present an economic simulation environment that features competitive pressures and market dynamics. We validate the simulation by showing that baseline tax systems perform in a way that is consistent with economic theory, including in regard to learned agent behaviors and specializations. Second, we show that AI-driven tax policies improve the trade-off between equality and productivity by 16% over baseline policies, including the prominent Saez tax framework. Third, we showcase several emergent features: AI-driven tax policies are qualitatively different from baselines, setting a higher top tax rate and higher net subsidies for low incomes. Moreover, AI-driven tax policies perform strongly in the face of emergent tax-gaming strategies learned by AI agents. Lastly, AI-driven tax policies are also effective when used in experiments with human participants.

The AI Economist

Improving Equality and Productivity with AI-Driven Tax Policies

salesforce research

https://www.youtube.com/watch?v=4iQUcGyQhdA&t=8s

DeepMind

2020-12-17

Open Problems in Cooperative AI

Allan Dafoe[1], Edward Hughes[2], Yoram Bachrach[2], Tantum Collins[2], Kevin R. McKee[2], Joel Z. Leibo[2], Kate Larson[2, 3] and Thore Graepel[2]

[1]Centre for the Governance of AI, Future of Humanity Institute, University of Oxford, [2]DeepMind, [3]University of Waterloo

Problems of cooperation—in which agents seek ways to jointly improve their welfare—are ubiquitous and important. They can be found at scales ranging from our daily routines—such as driving on highways, scheduling meetings, and working collaboratively—to our global challenges—such as peace, commerce, and pandemic preparedness. Arguably, the success of the human species is rooted in our ability to cooperate. Since machines powered by artificial intelligence are playing an ever greater role in our lives, it will be important to equip them with the capabilities necessary to cooperate and to foster cooperation.

References

- Lucian Busoniu, Robert Babuska, Bart De Schutter. A Comprehensive Survey of Multiagent Reinforcement Learning. In IEEE Transactions on Systems, Man and Cybernetics. Volume 38. Issue 2. March 2008.

- Kevin Leyton-Brown and Yoav Shoham. Multiagent Systems, Game-theoretic and Logical Foundations. Cambridge University Press. 2009.

- Karl Tuyls and Gerhard Weiss. Multiagent Learning: Basics, Challenges and Prospects. AI Magazine. Volume 33. Issue 3. 2012.

References

- Stefano V. Albrecht and Peter Stone. Autonomous Agents Modelling Other Agents: A Comprehensive Survey and Open Problems. Artificial Intelligence. Volume 258. 2018.

- Kaiqing Zhang, Zhuoran Yang and Tamer Basar. Multi-agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. 2021. arXiv:1911.10635v2.

- Sven Gronauer and Klaus Diepold. Multi-agent Reinforcement Learning: A Survey. Artificial Intelligence Review. 55:895-943. Springer. 2022.

References

- Karl Tulys and Peter Stone. Multiagent Learning Paradigms. In Francesco Belardinelli and Estefania Argente, editors, Multi-agent Systems and Agreement Technologies. Lecture Notes in Artificial Intelligence. Pages 3-21. Springer 2018.

- Micheal Wooldridge. An Introduction to MultiAgent Systems. Second Edition. Wiley. 2009.