



Master Reinforcement Learning 2022

Lecture 7: Multi-Agent

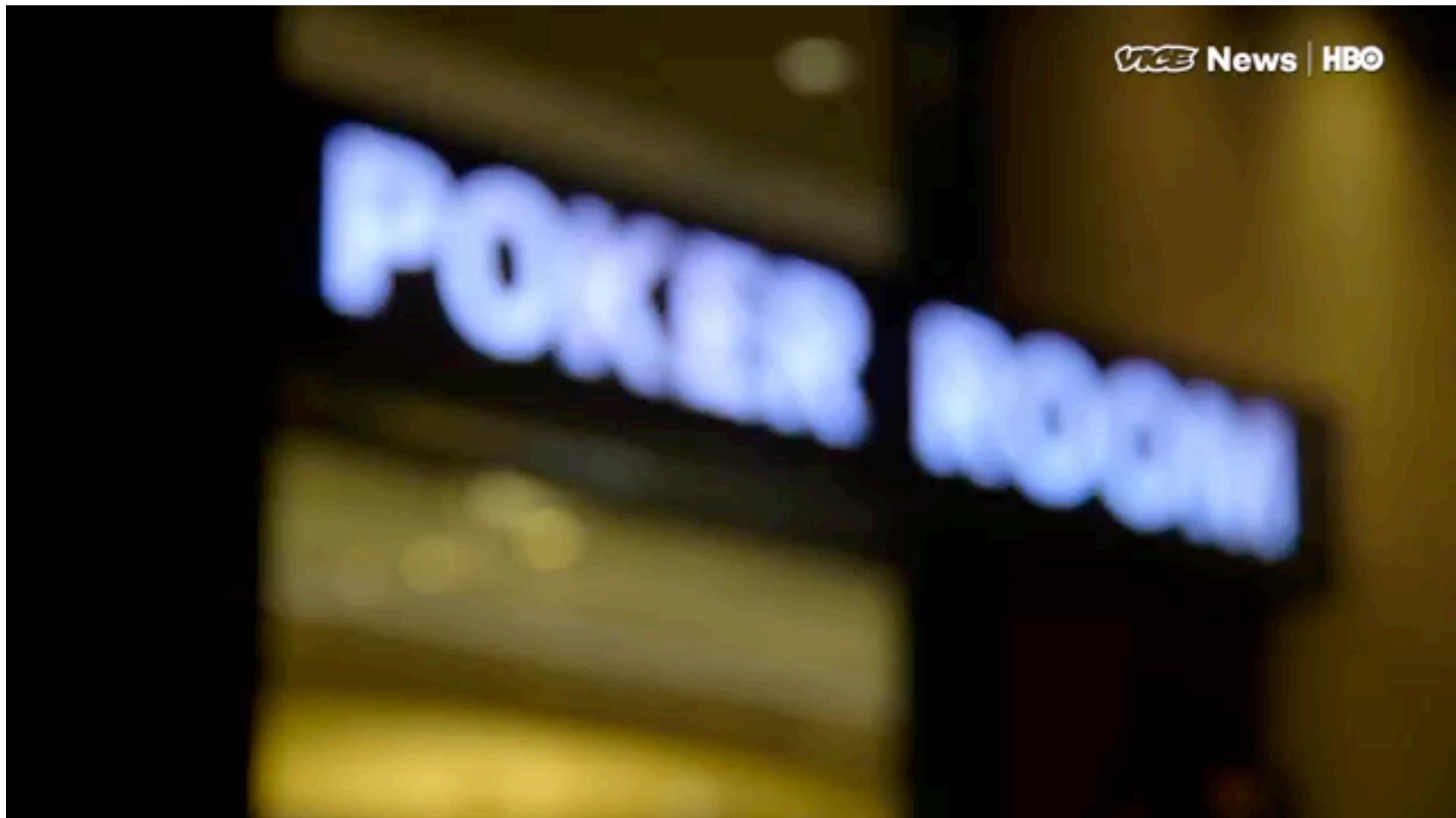
Aske Plaat



Different Approaches

- Model-free
 - Value-based [2,3]
 - Policy-based [4]
- Model-based
 - Learned [5]
 - Perfect; Two-Agent [6]
- Multi-agent [7]
- Hierarchical Reinforcement Learning (Sub-goals) [8]
- Meta Learning [9]

Motivation



Motivation

- Real-world decision making: model interaction

- Poker
- StarCraft
- Football



Overview

- 1: Competitive
 - 2: Cooperative
 - 3: Mixed
-
- 1: CFR
 - 2: Centr/Decentr, Opponent
 - 3: Evo, Swarm, Population Based Teams
-
- 1: Poker
 - 2: Hide and Seek
 - 3: Capture The Flag

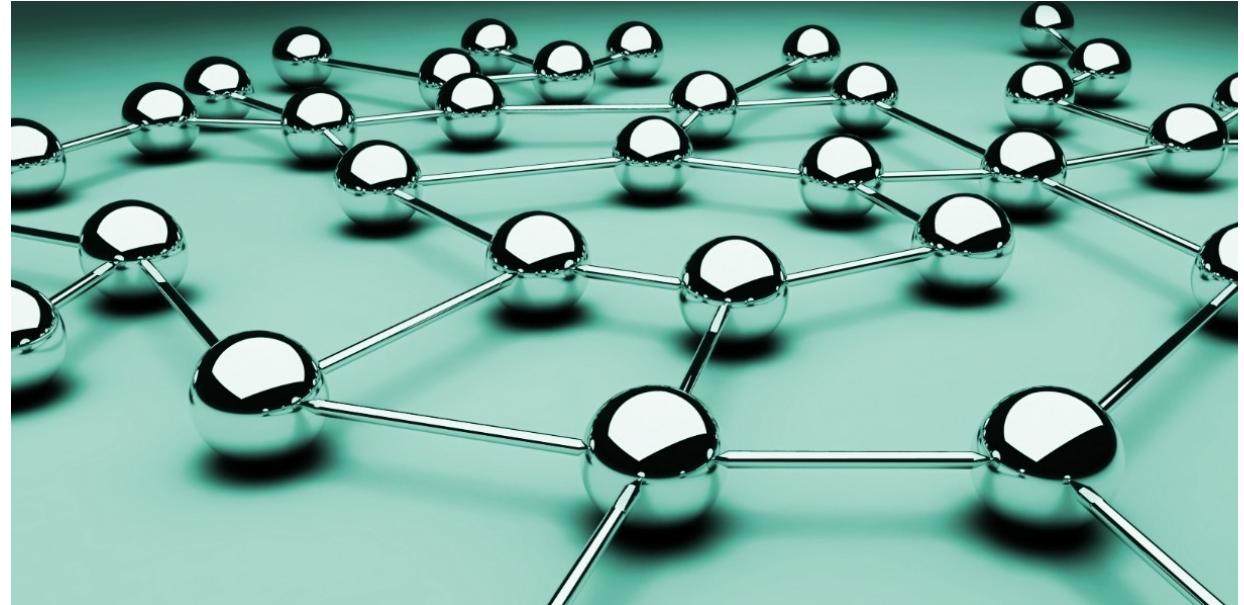
Social Behavior

- Modeling competition; egoism
- Modeling cooperation; altruism
- Emergent social behavior



Related Fields

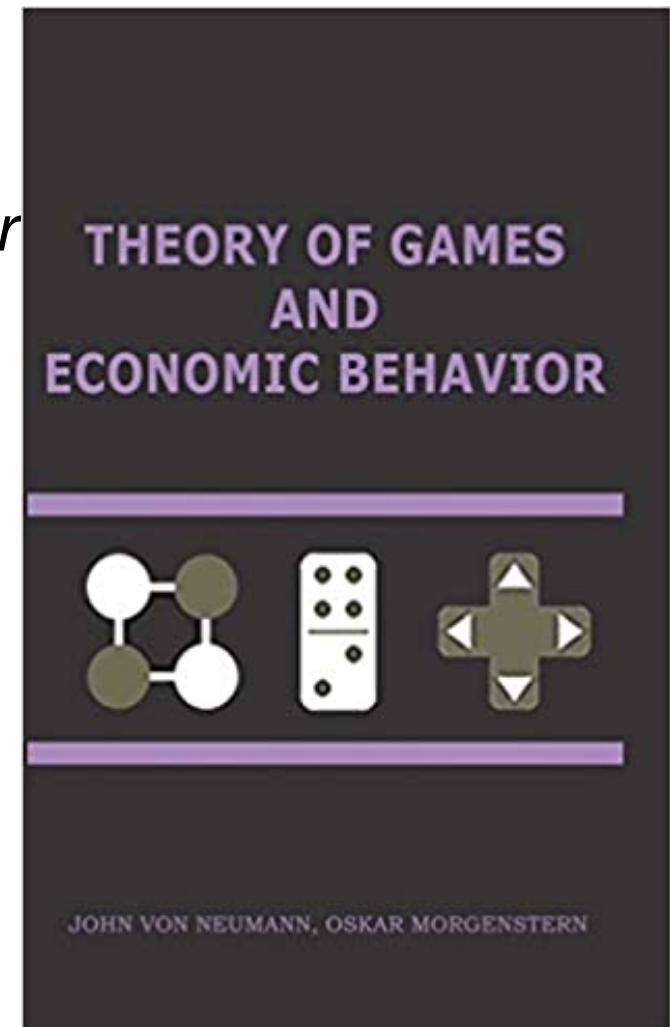
- Multi-agent Systems
- Swarm computing, Evolutionary Algorithms
- Complex Networks, Real World Networks

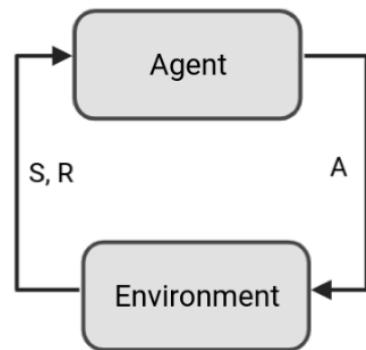


Multi-agent problems

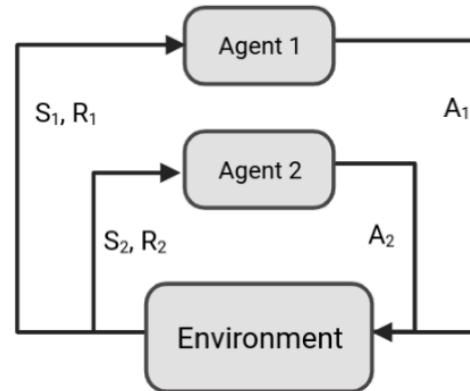
Game Theory

- Von Neumann & Morgenstern, 1944
Theory of Games and Economic Behavior
- MDP
- Partial information: POMDP

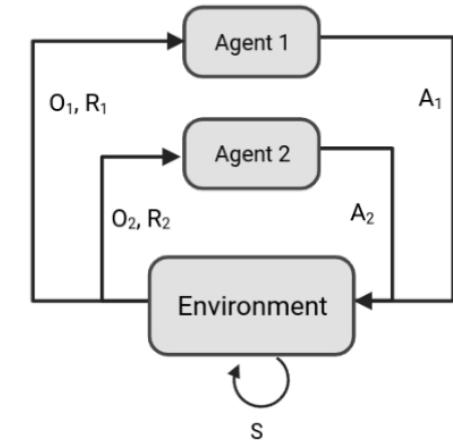




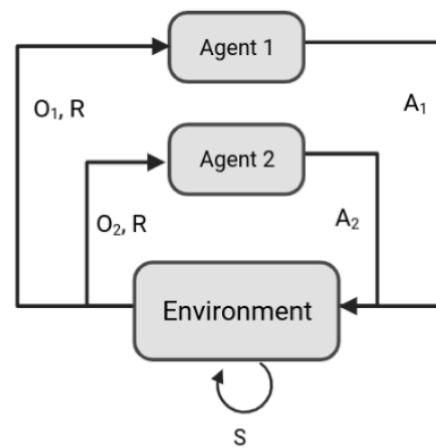
MDP



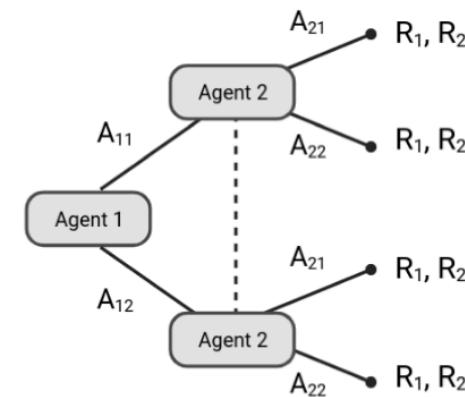
Markov Game



POMG



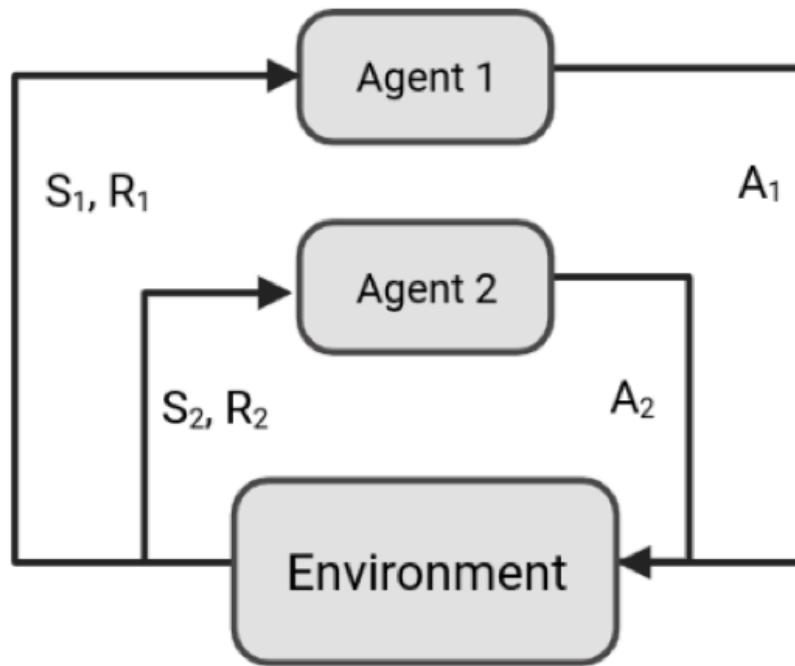
Dec-POMDP



Extensive Form

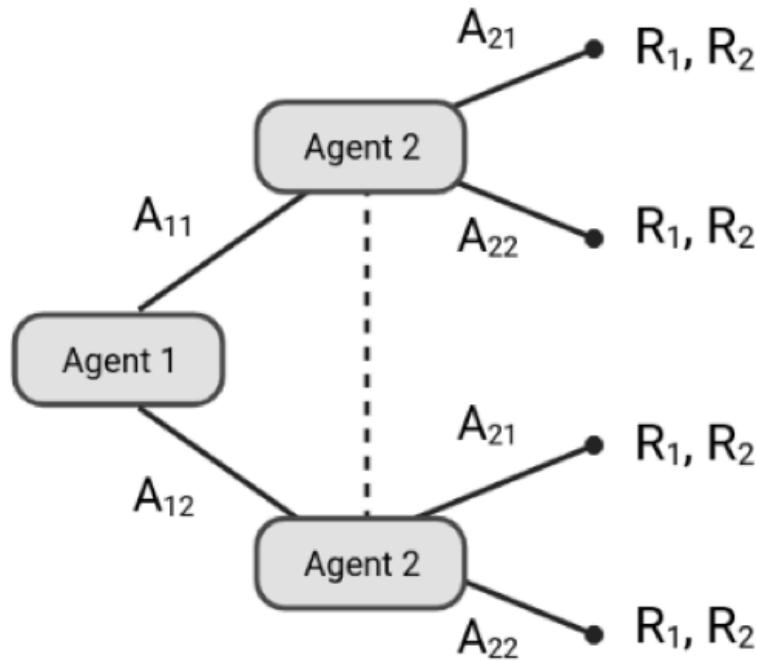
Fig. 2 Visual depiction of the main problem representations in multiagent reinforcement learning The MDP is the primary framework used in the single-agent setting. An agent is in some state S , performs action A , and receives a reward R from the environment. In partially observable environments, the agent cannot view the true state S and receives an observation O instead. For simplicity, all figures display the interaction between two agents $i = 1, 2$ but can be extended to more agents.

Stochastic Games



- Stochastic Games
- Markov Games

Extensive-form Games



- Imperfect information games
- Possible outcomes: information set

Competition

Competition

- Zero sum; win/loss

- John Nash:

The Nash equilibrium is point π^* from which in a non-collaborative setting none of the agents has any incentive to deviate.

- It is the optimal competitive strategy; each agent chooses best actions for themselves assuming others do the same



Nash equilibrium

- “Multi-agent minimax”
- The Nash-policy for an agent is its best-response strategy
- It is guaranteed to do no worse than tie against any opponent strategy
- For games of imperfect information the Nash equilibrium is an expected outcome

Nash equilibrium

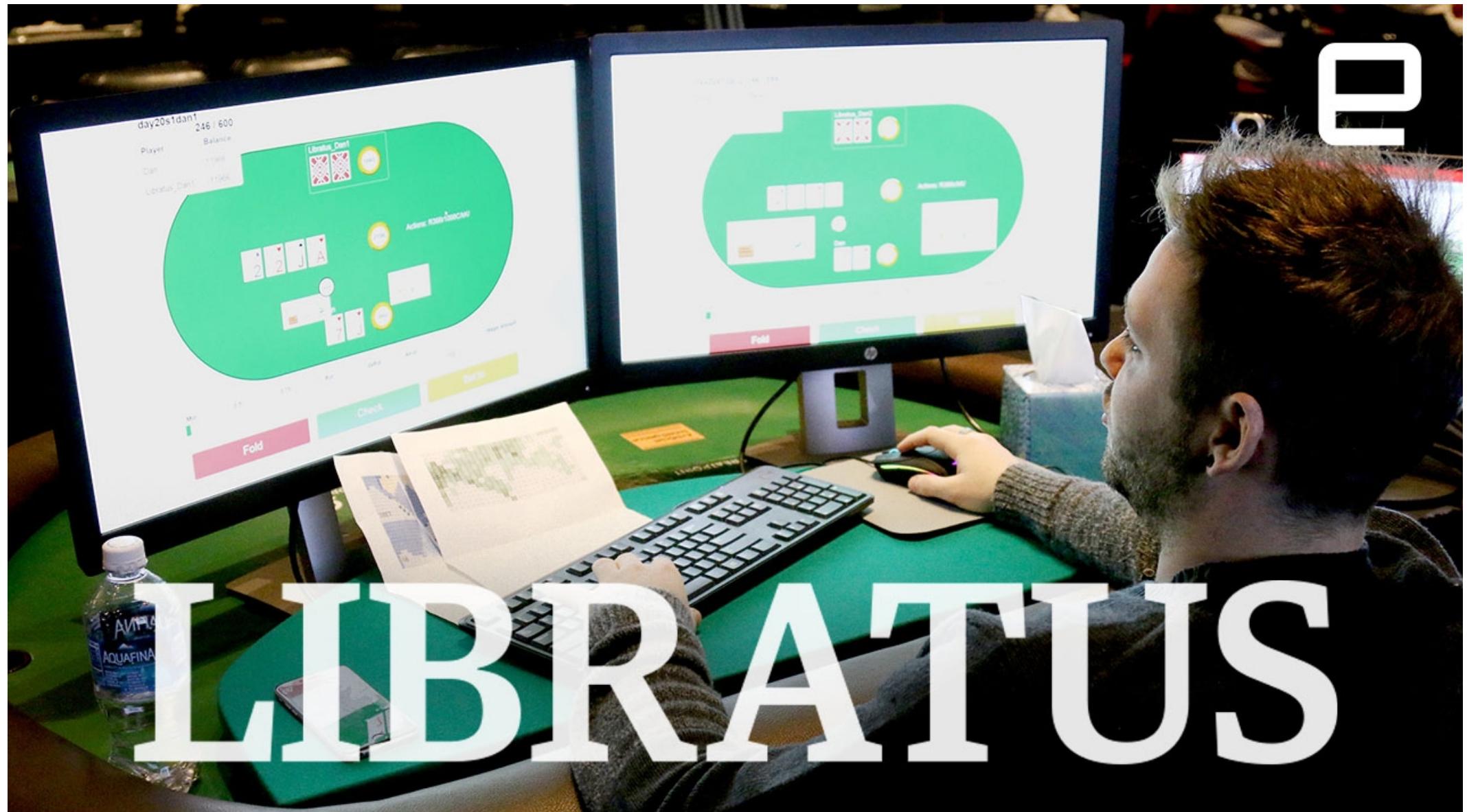
		Firm B
		Hold down output
		Increase output
Firm A	Hold down output	A gets \$1,000 B gets \$1,000
	Increase output	A gets \$200 B gets \$1,500
Increase output	Hold down output	A gets \$1,500 B gets \$200
	Increase output	A gets \$400 B gets \$400

← Nash

Counterfactual Regret Minimization

- Multi-agent, partial information, competition
- Algorithm: Counterfactual regret minimization
- Minimize the regret of not having taken the right action, playing many “what-ifs” (counterfactuals)
- CFR is probabilistic multi-agent version of competitive minimax
- Works quite well in Poker
- Complicated code, see paper

Poker



LIBRATUS

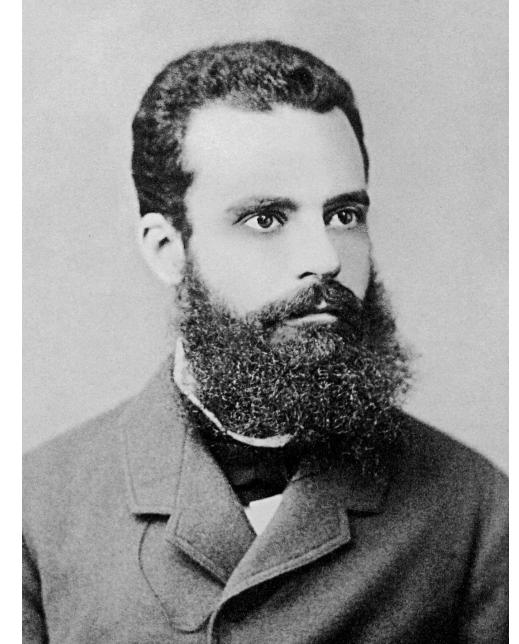
Pluribus



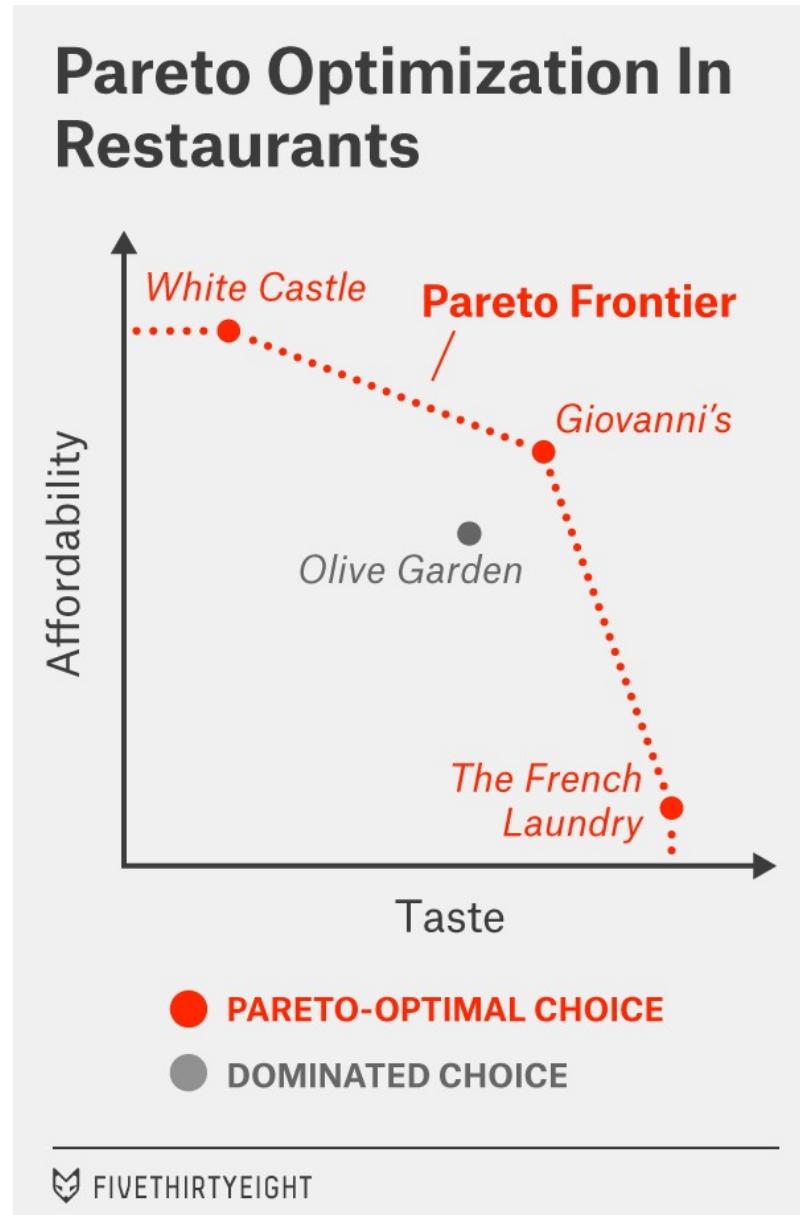
Cooperation

Cooperation

- Non zero sum; win/win
- Vilfredo Pareto
Pareto front is, in a cooperative setting, the combination of choices where no agent can be better off without at least making one other agent worse off
- It is the optimal cooperative strategy, the best outcome without hurting others.



Pareto front



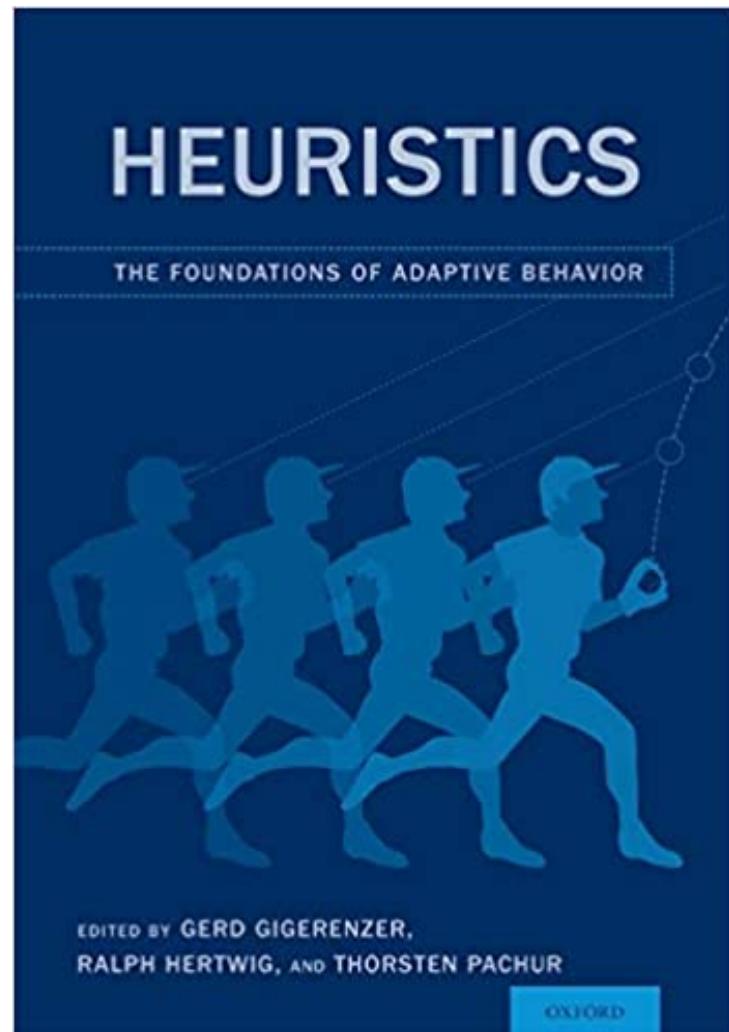
Cooperative Behavior

- Dealing with nonstationarity and partial observability can be done (ignored) by separate training, no communication
- Realism can be improved with Centralized Training/Decentralized Execution -> Centralized controller, or interaction graphs
- Active field of research; overview
 - Value based: VDN, QMIX
 - Policy based: COMA, MADDPG
 - Opponent modeling: DRON, LOLA
 - Communication: Diplomacy game
 - Psychology: Heuristics

Heuristics

SIMPLE
HEURISTICS
THAT MAKE US
SMART

GERD GIGERENZER, PETER M. TODD,
AND THE ABC RESEARCH GROUP



Emergent Cooperation

- [Baker, 2019]
- The agents can **move** by setting a force on themselves in the x and y directions as well as rotate along the z-axis.
- The agents can **see** objects in their line of sight and within a frontal cone.
- The agents can **sense** distance to objects, walls, and other agents around them using a lidar-like sensor.
- The agents can **grab and move** objects in front of them.
- The agents can **lock** objects in place. Only the team that locked an object can unlock it.

Hide and Seek



Mixed

- Prisoner's dilemma
- Iterated prisoner's dilemma
- Emerging social norms

Prisoner's Dilemma

		Confess	Silent
		Defect	Cooperate
Confess	(-5, -5)	(0, -10)	
Defect		<i>Nash</i>	
Silent	(-10, 0)	(-2, -2)	
Cooperate			<i>Pareto</i>

Iterated Prisoner's Dilemma

- You remember “opponent’s” behavior
- You will continue to meet your “opponents”
- Famous Experiment by Axelrod
- Rapoport introduced Tit for Tat
- You start being nice (Cooperating) and then do what the other did the previous round

Tit for Tat

Defector	Tit For Tat	Cooperator
Always Rats out	Starts not ratting then mimics other player	Never Rats out
D vs D	T vs T	C vs C
Both always rat, gain moderate points	Both never rat, gain many points	Both never rat, gain many points
D vs T	D vs C	C vs T
After first round both always rat, gain moderate points (D slightly more)	D always rats and gains maximum points, C never rats and gains no points	Both never rat, gain many points

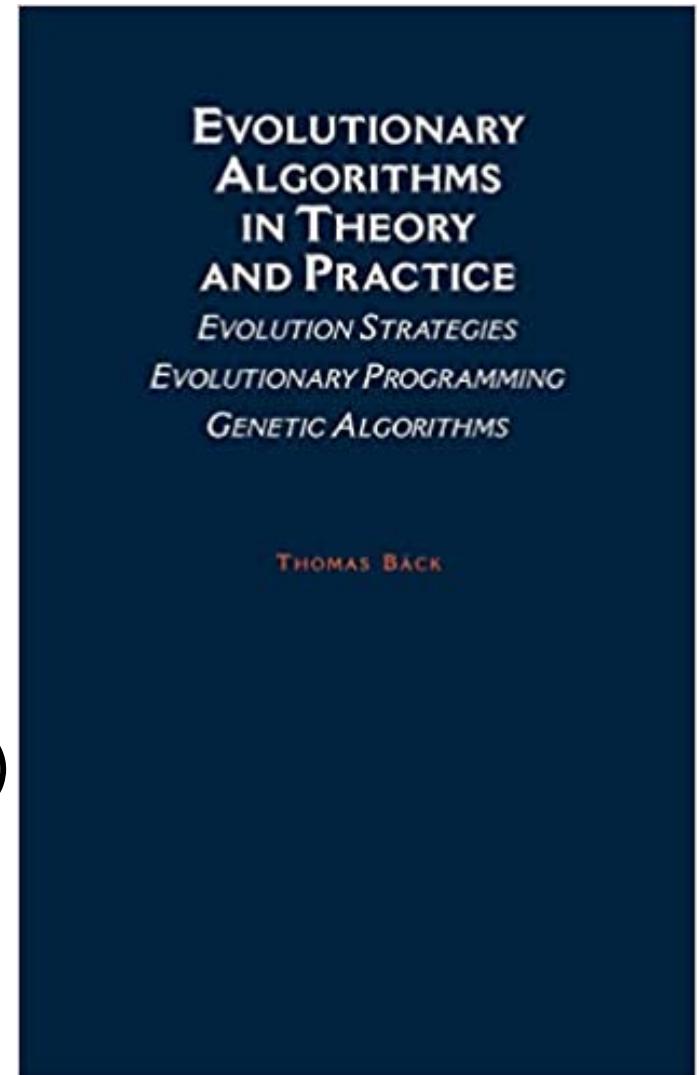
Algorithms

Challenges

- Partial Observability -> Large State Space
(Information sets)
- Nonstationary Environments -> Large State Space
(Calculate all configurations)
- Multiple Agents -> Large State Space
(Esp. with simultaneous actions)

Evolutionary Approaches

- Evolutionary Algorithms
- Swarm Computing
- Population based training (teams, HRL)



Evolutionary Framework

Algorithm 7.1 Evolutionary Framework [36]

- 1: Generate the initial population randomly
 - 2: **repeat**
 - 3: Evaluate the fitness of each individual of the population
 - 4: Select the fittest individuals for reproduction
 - 5: Through crossover and mutation generate new individuals
 - 6: Replace the least fit individuals by the new individuals
 - 7: **until** terminated
-

Evo

Multi-Agent Reinforcement Learning	Evolutionary Computation
agent	individual
some	many
all agents	population
environment	problem
reward	fitness
policy	genes
adaptation	mutation and combination
time step	generation
feedback	selection

- Highly parallel
- Multi-agent population based optimization
- Single-agent deep network policy optimization instead of backpropagation
- Single fitness function, determines cooperation or competition

Swarm Intelligence Algorithms

A Tutorial



Edited by
Adam Slowik

Ant Colony Optimization

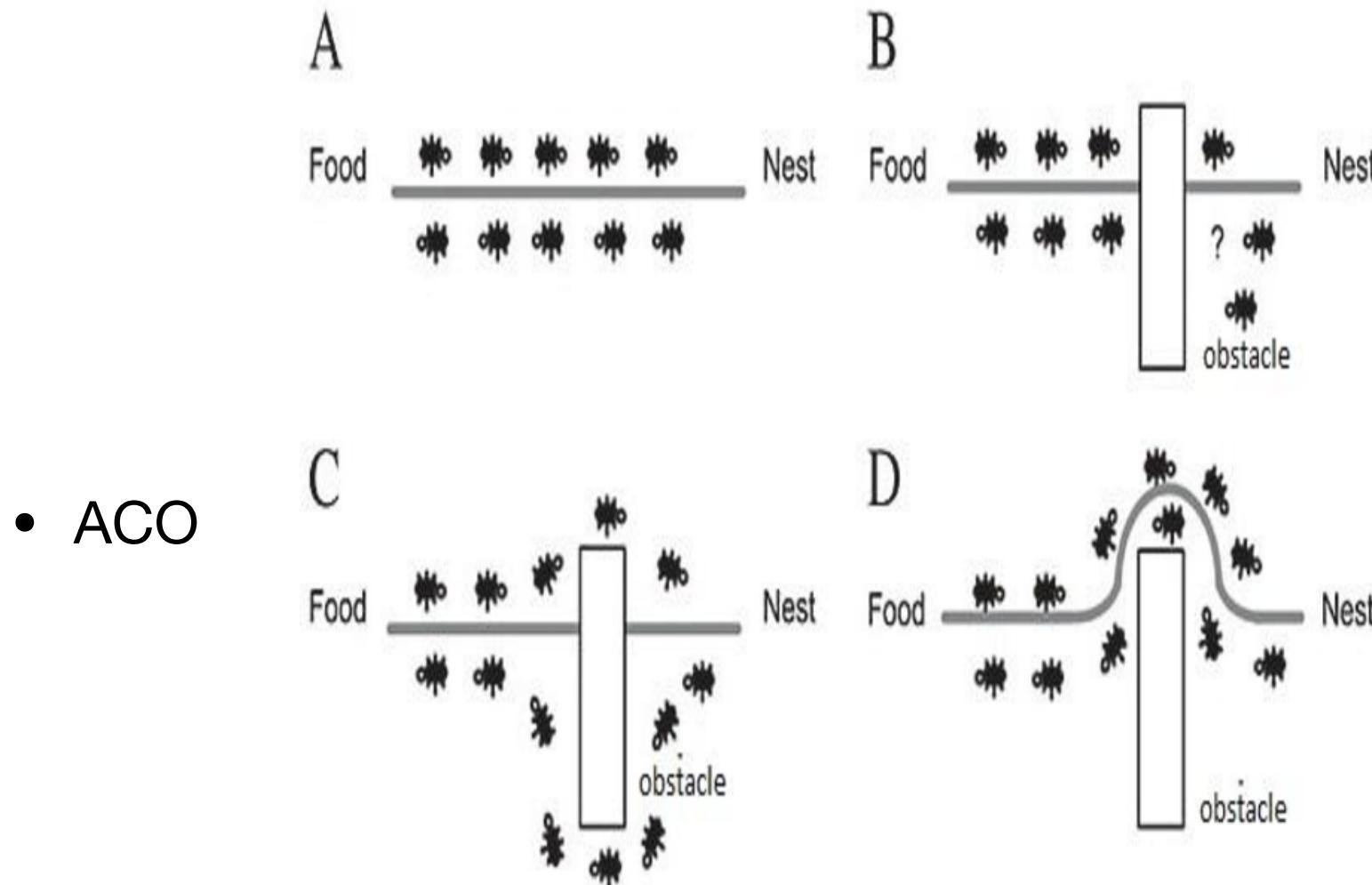


Fig 1.

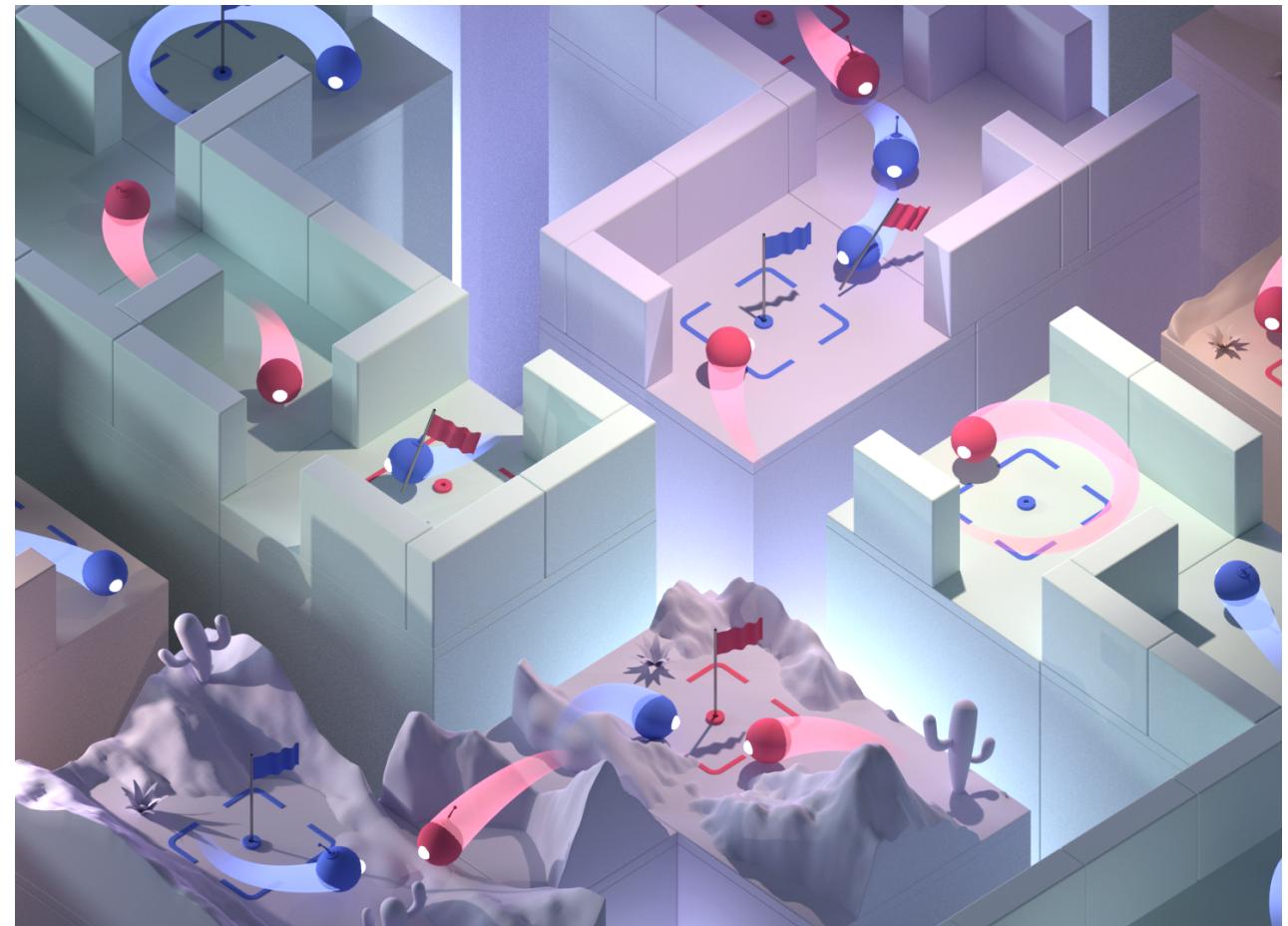
A: Ants in a pheromone trail between nest and food.

B: an obstacle interrupts the trail.

C: Ants find two paths to go around the obstacle

Population-based training

- Teams
- Hierarchical
- Cooperation, competition
- Within Teams, between teams
- Blends RL and Evo



Population-based training

Algorithm 7.2 Population Based Training [352]

```
procedure TRAIN( $\mathcal{P}$ )                                ▷ initial population  $\mathcal{P}$ 
    for  $(\theta, h, p, t) \in \mathcal{P}$  (asynchronously in parallel) do
        while not end of training do
             $\theta \leftarrow \text{step}(\theta | h)$           ▷ one step of optimisation using hyperparameters  $h$ 
             $p \leftarrow \text{eval}(\theta)$                 ▷ current model evaluation
            if ready( $p, t, \mathcal{P}$ ) then
                 $h', \theta' \leftarrow \text{exploit}(h, \theta, p, \mathcal{P})$   ▷ use the rest of population for improvement
                if  $\theta \neq \theta'$  then
                     $h, \theta \leftarrow \text{explore}(h', \theta', \mathcal{P})$       ▷ produce new hyperparameters  $h$ 
                     $p \leftarrow \text{eval}(\theta)$                       ▷ new model evaluation
                end if
            end if
            update  $\mathcal{P}$  with new  $(\theta, h, p, t + 1)$            ▷ update population
        end while
    end for
    return  $\theta$  with the highest  $p$  in  $\mathcal{P}$ 
end procedure
```

CTF



**TWO MINUTE
PAPERS**

SUPERHUMAN QUAKE 3 AI TEAM

Disclaimer: I was not part of this research project, I am merely providing commentary on this work.

StarCraft



StarCraft

- Real Time Strategy
- 10^{1685}
- AlphaStar
- Population based multi agent methods

The screenshot shows a web browser window on a Mac OS X desktop. The address bar at the top says "nature.com". Below the address bar, there are several tabs and links: "Course documents...", "mountain car - Goo...", "(16) (PDF) Adaptive...", "feedback acoustic...", "(1) Als eerste rijden...", "pluribus poker - Goog...", "alphastar ai - Goog...", "Grandmaster level i...", and a "+" button. Underneath these, there are three buttons: "Supplements 2", "Figures 12", and "Metrics 3". The main content area is titled "Article" and features the title "Grandmaster level in StarCraft II using multi-agent reinforcement learning" in large bold letters. Below the title, the authors listed are Oriol Vinyals^{1,2*}, Igor Babuschkin^{1,2}, Wojciech M. Czarnecki^{1,2}, Michaël Mathieu^{1,2}, Andrew Dudzik^{1,2}, Junyoung Chung^{1,2}, David H. Choi^{1,2}, Richard Powell^{1,2}, Timo Ewalds^{1,2}, Petko Georgiev^{1,2}, Junhyuk Oh^{1,2}, Dan Horgan^{1,2}, Manuel Kroiss^{1,2}, Ivo Danihelka^{1,2}, Aja Huang^{1,2}, Laurent Sifre^{1,2}, Trevor Cai^{1,2}, John P. Agapiou^{1,2}, Max Jaderberg¹, Alexander S. Vezhnevets¹, Rémi Leblond¹, Tobias Pohlen¹, Valentin Dalibard¹, David Budden¹, Yury Sulsky¹, James Mollov¹, Tom L. Paine¹, Caglar Gulcehre¹, Ziyu Wang¹, Tobias Pfaff¹, Yuhuai Wu¹, Roman Ring¹, Dani Yogatama¹, Dario Wünsch², Katrina McKinney¹, Oliver Smith¹, Tom Schaul¹, Timothy Lillicrap¹, Koray Kavukcuoglu¹, Demis Hassabis¹, Chris Apps^{1,2} & David Silver^{1,2*}. Below the authors, a short summary reads: "Many real-world applications require artificial agents to compete and coordinate with other agents in complex environments. As a stepping stone to this goal, the domain of StarCraft has emerged as an important challenge for artificial intelligence research, owing to its iconic and enduring status among the most difficult professional esports and its relevance to the real world in terms of its raw complexity and multi-agent challenges. Over the course of a decade and numerous competitions^{1–3}, the strongest agents have simplified important aspects of the game, utilized superhuman capabilities, or employed hand-crafted sub-systems⁴. Despite these advantages, no previous agent has come close to matching the overall skill of top StarCraft players. We chose to address the challenge of StarCraft using general-purpose learning methods that are in principle applicable to other complex domains: a multi-agent reinforcement learning algorithm that uses data from both human and agent games within a diverse league of continually adapting strategies and counter-strategies, each represented by deep neural networks^{5,6}. We evaluated our agent, AlphaStar, in the full game of StarCraft II, through a series of online games against human players. AlphaStar was rated at Grandmaster level for all three StarCraft races and above 99.8% of officially ranked human players."

Questions?

