Double oracle-based methods



15 888 **Computational Game Solving** (Fall 2026)

Ioannis Anagnostides

Today’s lecture: making RL game-theoretically sound

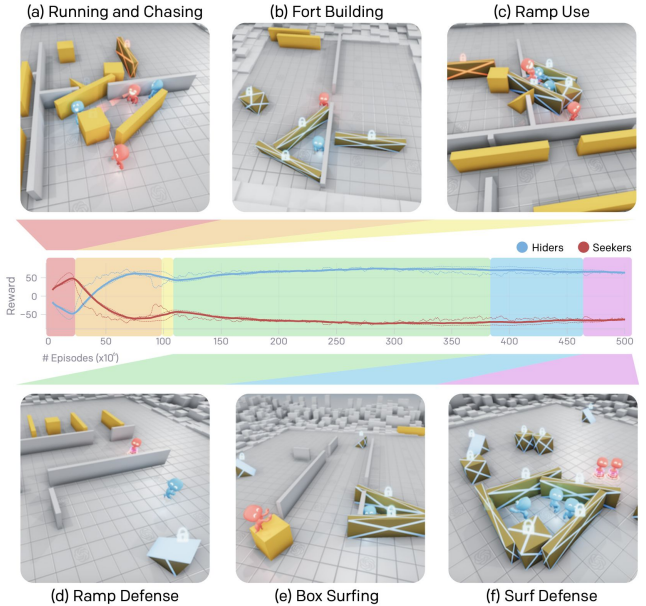
- Double oracle-based methods for solving zero-sum games
  - The original method by McMahan et al. (ICML 2003)
  - Anytime double oracle (McAleer et al., 2022)
  - Policy-space response oracles (PSRO) and variations
    - Anytime PSRO
    - Pipeline PSRO
    - Self-play PSRO
- Applications in solving large real-time strategy games
  - DeepMind’s AlphaStar for Starcraft
  - OpenAI Five for Dota 2

Making RL game-theoretically sound

What if we let each player in a zero-sum game employ an RL algorithm?

⚠ Not guaranteed to converge

Our goal: game-theoretically sound ways to use single-agent RL



Double oracle method

- Double oracle maintains throughout a **restricted** set of pure strategies
- Finds a **Nash equilibrium** of the game induced by the restricted strategies
- Computes a (pure) **best response** strategy for each player
- Adds to the set of restricted strategies the best responses, unless the best-response gap was zero

From McAleer et al. (2022)

Algorithm 1 Double Oracle (McMahan et al., 2003)

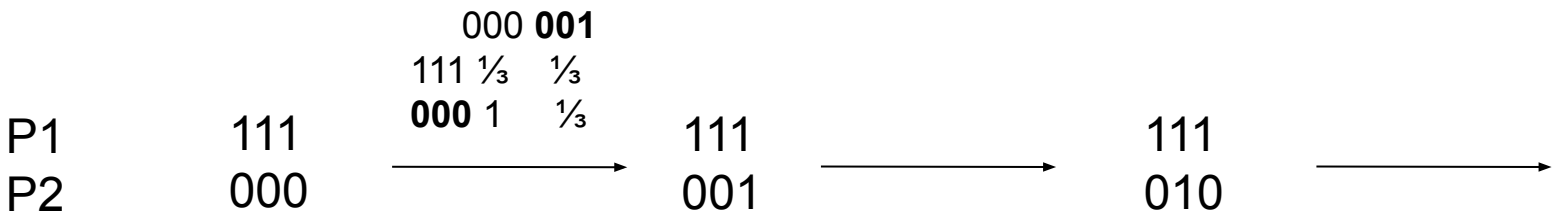
Result: Nash Equilibrium
Input: Initial population Π^0
repeat {for $t = 0, 1, \ldots$}
  $\pi^r \leftarrow$ NE in game restricted to strategies in Π^t
  for $i \in \{1, 2\}$ **do**
    Find a best response $\beta_i \leftarrow \mathbb{BR}_i(\pi^r_{-i})$
    $\Pi_i^{t+1} \leftarrow \Pi_i^t \cup \{\beta_i\}$
until No novel best response exists for either player
Return: π^r

Properties of double oracle method

- DO is guaranteed to end up at an exact Nash equilibrium
- May converge quickly in some games, but in the worst-case it needs to expand all pure strategies
- In extensive-form games, it was recently shown that it can take **exponentially many iterations** (Zhang and Sandholm, IJCAI 2024)
- Relies fundamentally on best responses, just like fictitious play

Slow convergence in extensive-form games

- Chance selects one of k states uniformly at random
- Both players observe the outcome and simultaneously choose 0 or 1
- At s1, they play matching pennis—P1 wins if the actions match
- At every other state, Player 2 wins when (a1, a2) = (0, 1)
- The winner receives +1, the loser -1
- Under **adversarial selection**, double oracle can take **exponential time**



Zhang and Sandholm (IJCAI 2024)

Outlook of the double oracle method

+ Can sometimes terminate after a small number of iterations
+ Compatible with deep RL (more on that soon)
+ Good performance in very large games

− Exponential worst-case performance, even in 2p0s games
− Requires finding a Nash equilibrium in a series of restricted games
− Non-monotonic exploitability

Outlook of the double oracle method

+ Can sometimes terminate after a small number of iterations
+ Compatible with deep RL (more on that soon)
+ Good performance in very large games

− Exponential worst-case performance, even in 2p0s games
− Requires finding a Nash equilibrium in a series of restricted games
− Non-monotonic exploitability

Anytime double oracle

- An anytime guarantee means that one can safely stop the algorithm at any moment during training
- Highly desirable in practice
- **Anytime DO**, introduced by McAleer et al. (2022), addresses this issue
- It is a variant of DO that guarantees nonincreasing exploitability
- The key difference is in the **definition of the restricted game**

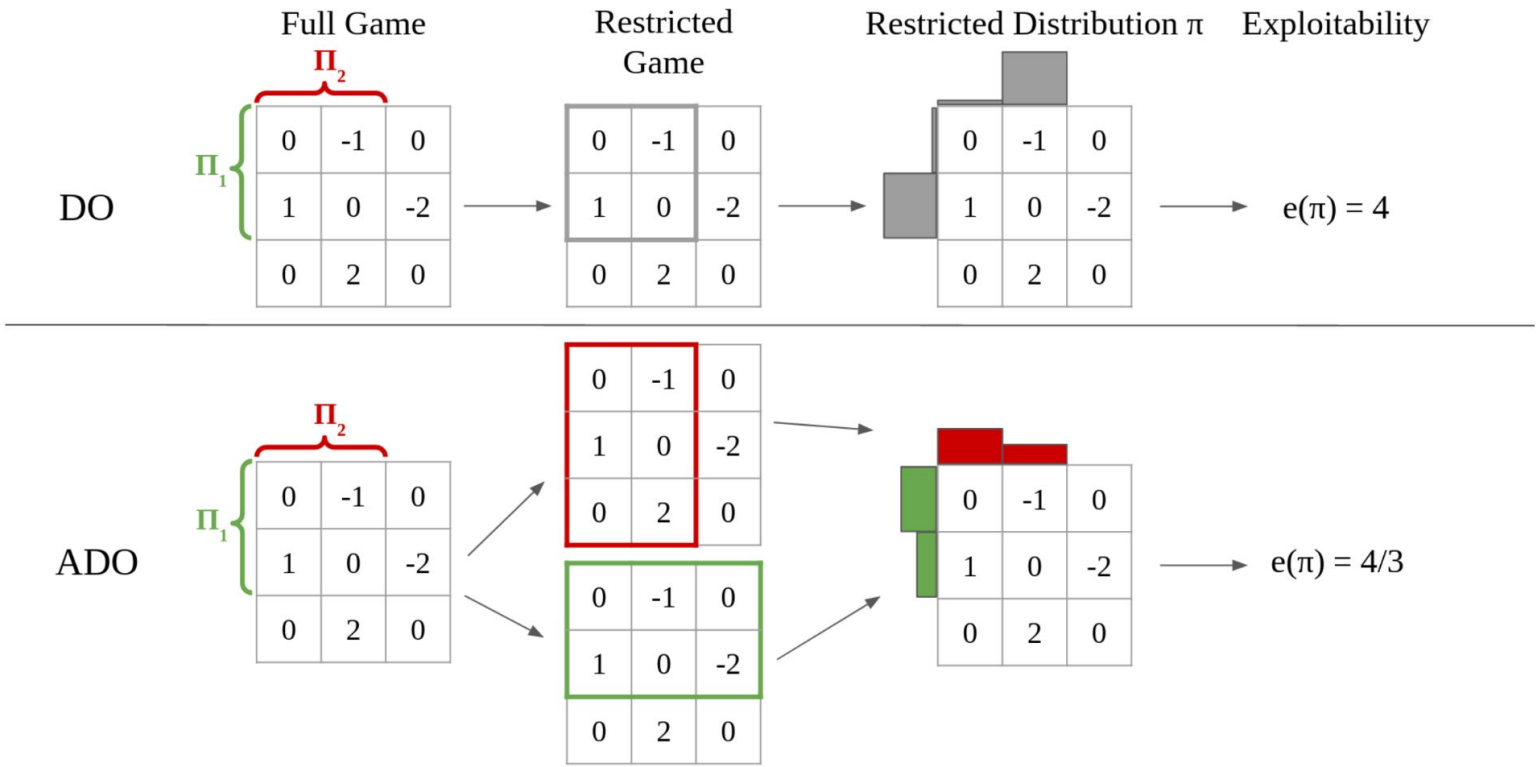
From McAleer et al. (2022)

Algorithm 2 Anytime Double Oracle (ADO)

Result: Nash Equilibrium
Input: Initial population Π^0
repeat {for $t = 0, 1, \ldots$}
  $\pi_i^r \leftarrow$ NE in restricted game G^i (eq. (1)), for $i \in \{1, 2\}$
  for $i \in \{1, 2\}$ **do**
    Find a novel best response $\beta_i \leftarrow \mathbb{BR}_i(\pi^r_{-i})$
    $\Pi_i^{t+1} = \Pi_i^t \cup \{\beta_i\}$
until No novel best response exists for either player
Return: π^r

Anytime double oracle

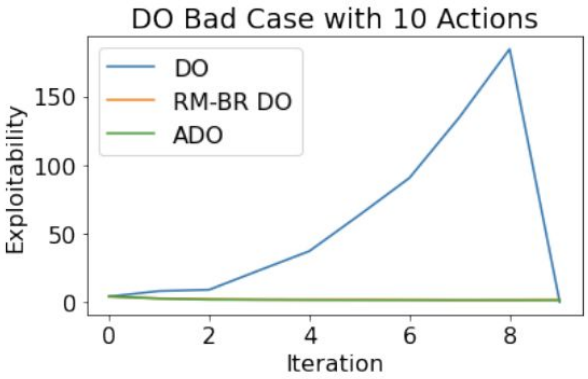
From McAleer et al. (2022)

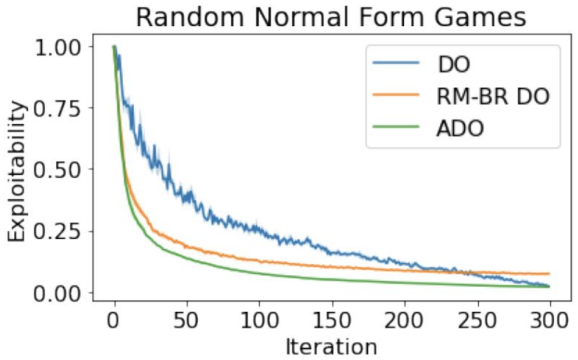


ADO + RM-BR

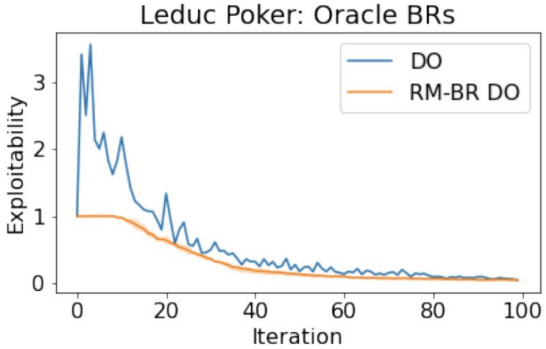
- Unlike DO, requires solving, even in the first iterations, a game in which **one player** has a large number of strategies
- How can we solve such games in a scalable way?
- Regret minimization + best-response (RM+BR) (recall homework 1) only requires a best-response oracle for the side with a large number of actions
- The regret *doesn't depend on the action set of the large player*!
- The resulting algorithm is called **RM-BR DO**; it's an approximate version of Anytime DO (McAleer et al. 2022)

ADO and RM-BR DO in action





(a) Random Normal Form Games



(b) Leduc with Oracle Best Responses

Policy-space response oracles (PSRO)

- PSRO is a conceptual generalization of double oracle and fictitious play
- It has a meta-solver, which outputs a meta-strategy of a restricted game
  - Nash equilibrium in DO
  - Uniform averages in fictitious play
- For each player, it finds a best response to the meta-strategy
- Approximate best responses are computed through RL

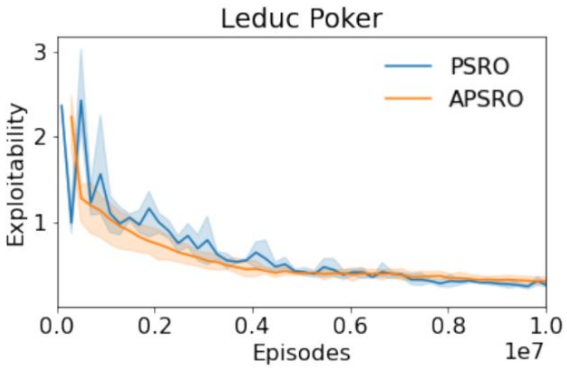
Lanctot et al. (NeurIPS 2017)

Algorithm 1: Policy-Space Response Oracles

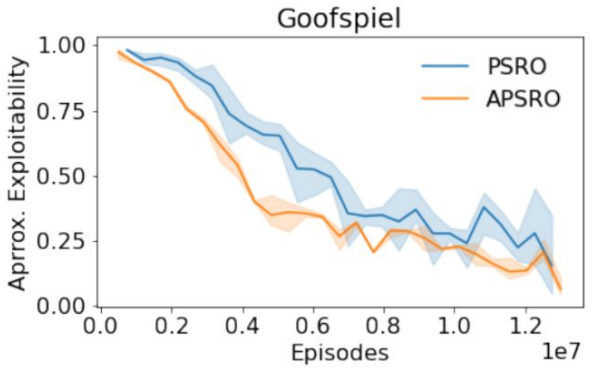
input : initial policy sets for all players Π
Compute exp. utilities U^{Π} for each joint $\pi \in \Pi$
Initialize meta-strategies $\sigma_i = \text{UNIFORM}(\Pi_i)$
while *epoch* e *in* $\{1, 2, \cdots\}$ **do**
  for *player* $i \in [[n]]$ **do**
    for *many episodes* **do**
      Sample $\pi_{-i} \sim \sigma_{-i}$
      Train oracle π_i' over $\rho \sim (\pi_i', \pi_{-i})$
    $\Pi_i = \Pi_i \cup \{\pi_i'\}$
  Compute missing entries in U^{Π} from Π
  Compute a meta-strategy σ from U^{Π}
Output current solution strategy σ_i for player i

Anytime PSRO experiments

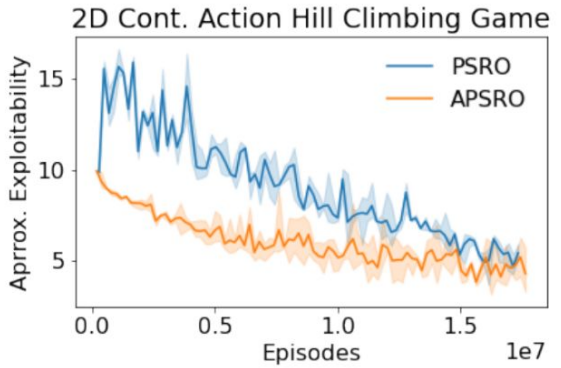
From McAleer et al. (2022)



(a) Leduc with DDQN BRs



(b) Goofspiel with DDQN BRs



(c) Continuous-Action Hill-Climbing Game

Self-play PSRO

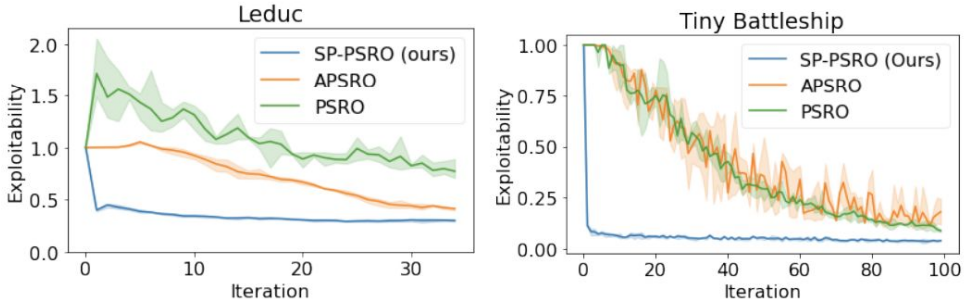
From McAleer et al. (ICLR 2024)

Algorithm 3 Self-Play PSRO

Result: Approximate Nash Equilibrium
Input: Initial population $\Pi^0 \cup \{\nu_i\}$
while Not Terminated $\{t = 0, 1, \ldots\}$ **do**
  Initialize new strategy ν_i arbitrarily
  Initialize π_i^r to uniform over $\Pi_i^t \cup \{\nu_i\}$ for $i \in \{1, 2\}$
  for $i \in \{1, 2\}$ **do**
    for n iterations **do**
      for m iterations **do**
        Update policy β_{-i} toward $\mathbb{BR}_{-i}(\pi_i^r)$
        Update new strategy ν_i toward $\mathbb{BR}_i(\beta_{-i})$
      Update π_i^r via regret minimization vs. β_{-i}
  $\Pi_i^{t+1} = \Pi_i^t \cup \{\beta_i, \bar{\nu}_i\}$ for $i \in \{1, 2\}$
Return: π^r

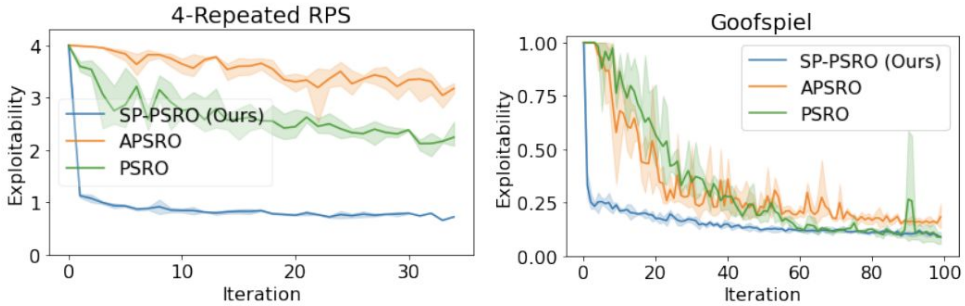
Self-play PSRO in action

From McAleer et al. (2024)



(a) Leduc Poker

(b) Battleship



(c) Repeated RPS

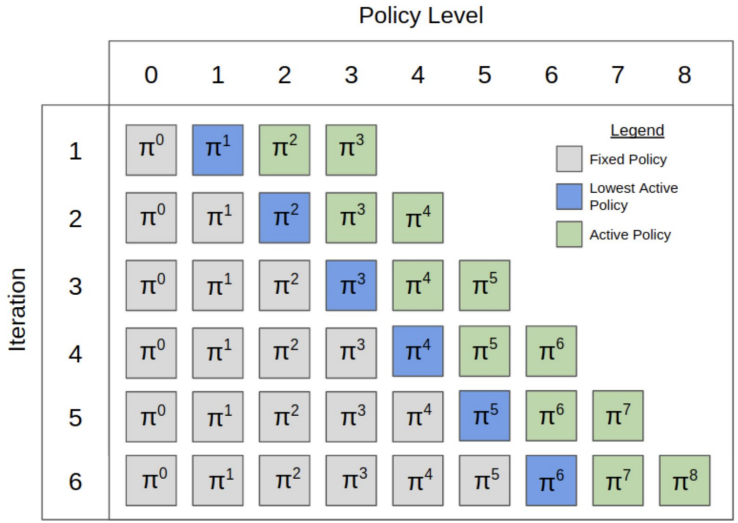
(d) Goofspiel

Figure 4: Extensive-form games with tabular Q-learning best responses

Pipeline PSRO

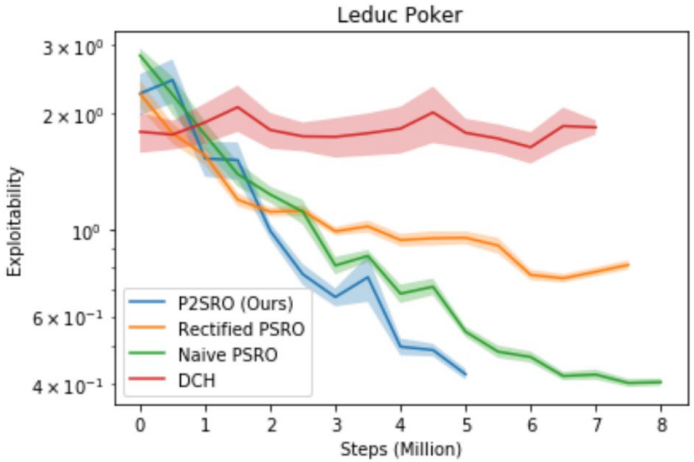
From McAleer et al. (2020)

- How can **parallel** workers improve PSRO?
- Pipeline PSRO (**P2SRO**), introduced by McAleer et al. (NeurIPS 2020), exploits parallelism in a sound way
- The lowest-level active policy plays against the meta NE of the lower-level fixed policies
- Each additional active policy plays against the meta Nash equilibrium of the fixed and training policies in levels below it
- Once the lowest active policy plateaus, it becomes fixed, a new active policy is added, and the next active policy becomes the lowest active policy

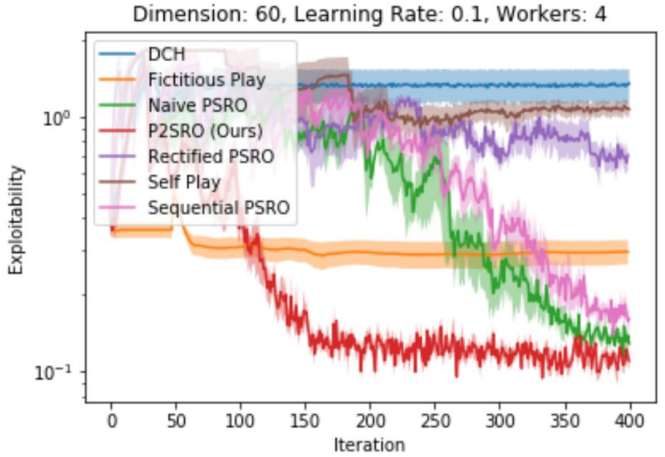


P2SRO in action

From McAleer et al. (2020)



(a) Leduc poker



(b) Random Symmetric Normal Form Games

P2SRO in Barrage Stratego

From McAleer et al. (2020)

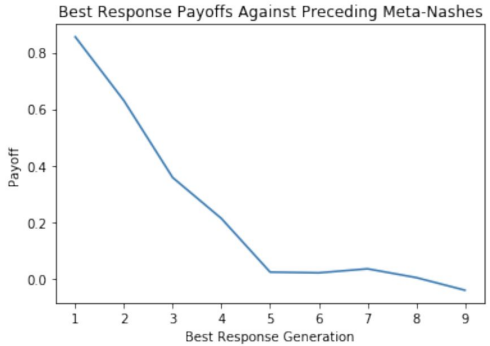
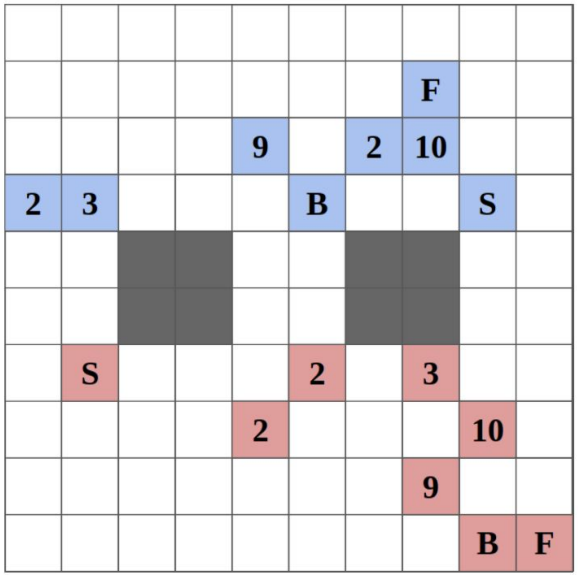


Figure 3: Barrage Best Response Payoffs Over Time

Name	P2SRO Win Rate vs. Bot
Asmodeus	81%
Celsius	70%
Vixen	69%
Celsius1.1	65%
All Bots Average	**71%**

Table 1: Barrage P2SRO Results vs. Existing Bots



AlphaStar

- *StarCraft* was an important milestone in AI research due to the game's real-time play, partial observability, large action space, and long-term planning
- Each side begins with workers and a base, gathers resources, and decides how much to invest in economy, technology, and military
- The most common variant is a 1vs1 tournament played over 5 games



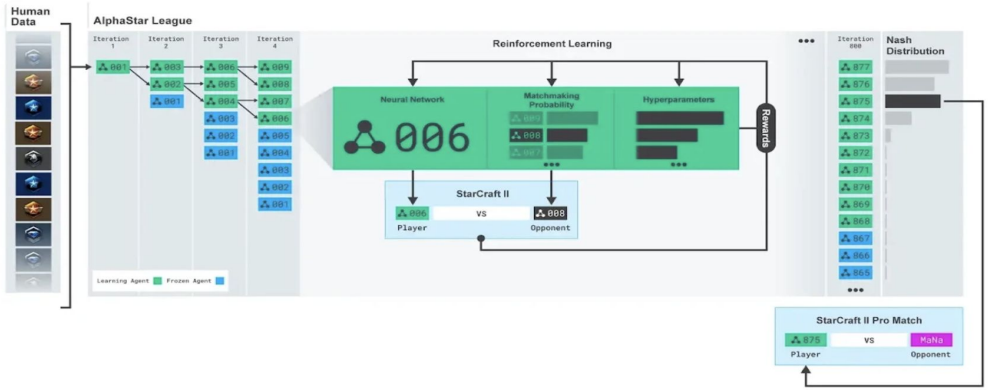
AlphaStar: fair comparison

- A computer can issue commands very quickly
  - Humans are constrained by the mechanical limits of computer mice
  - View of the map is dictated by the in-game camera
- Winning could reflect mechanical advantage
- AlphaStar uses action-rate limits and camera restrictions



From DeepMind blog

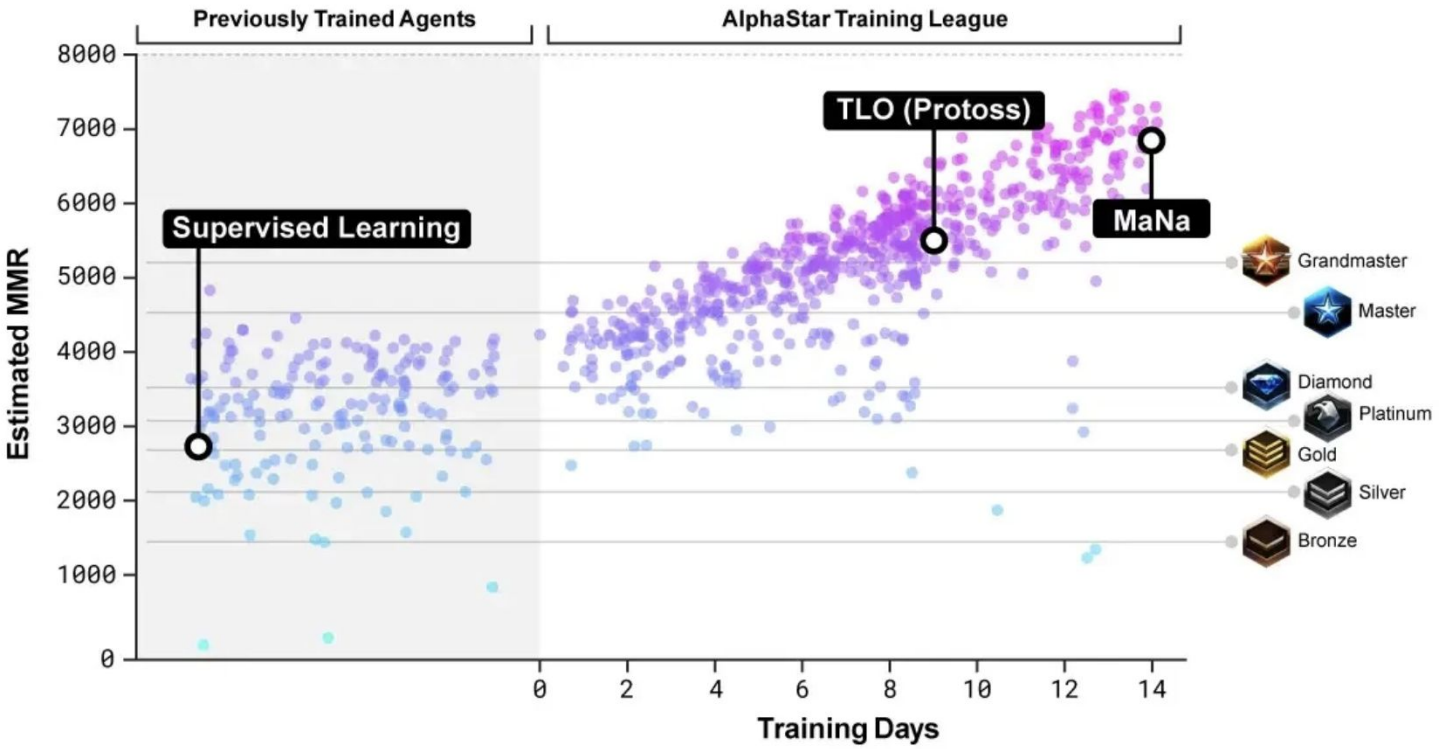
AlphaStar League



- League training: an algorithm based on a league in which agents, or competitors, play games against each other
- Agents are initially trained from human game replays (supervised learning)
- This is essential to solve the exploration problem: discovering good new strategies in a game like StarCraft is like finding a needle in a haystack
    - Learning some type of attack requires the right **sequence**: building the right structures and producing flying units, so it only pays off once the entire sequence succeeds

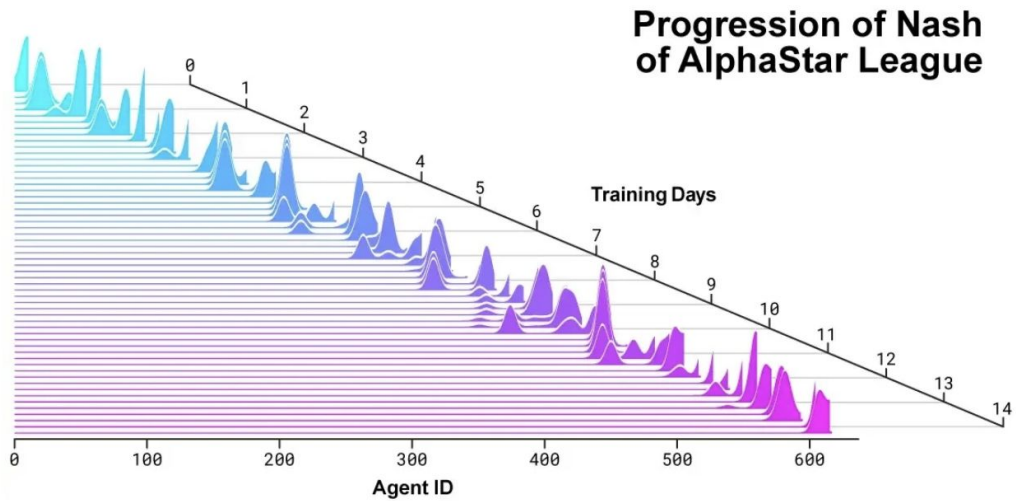
From DeepMind blog

Training progression



From DeepMind blog

Training progression



The Nash distribution over competitors as the AlphaStar league progressed and new competitors were created. The Nash distribution, which is the least exploitable set of complementary competitors, weights the newest competitors most highly, demonstrating continual progress against all previous competitors.

Population training (PSRO-type)

- **Main agents** utilize a *prioritized fictitious self-play (PFSP)* mechanism that adapts the mixture probabilities proportionally to the win rate of each opponent against the agent => more likely to play against tougher opponents
  - More practice against opponents we struggle against
- **Main exploiters** play only against the current iteration of main agents. Their purpose is to identify potential exploits in the main agents
- **League exploiter** agents use a similar PFSP mechanism to the main agents, but are not targeted by main exploiter agents. Their purpose is to find systemic weaknesses of the entire league

Evaluation

- AlphaStar Final played above 99.8% of ranked human players and at Grandmaster level
- Evaluation used anonymous accounts, so is does not directly test whether opponents could repeatedly study and exploit it
- AlphaStar Supervised placed above 84% of human players, and “shows the effectiveness of supervised learning” (?)
- On the flip side, AlphaStar was found to be exploitable: high playing strength does not imply low exploitability
- AlphaStar had a mechanism to enforce APM limits

OpenAI Five: Dota 2

- Dota 2 is a popular real-time team zero-sum game
- Two teams of five heroes compete to destroy the opposing team's main building
- Each team can be viewed as one strategic player, but the five heroes must coordinate their actions
    - A bad outcome for one hero can help the team win the challenge
- Presents a similar set of challenges to StarCraft



OpenAI Five: Dota 2

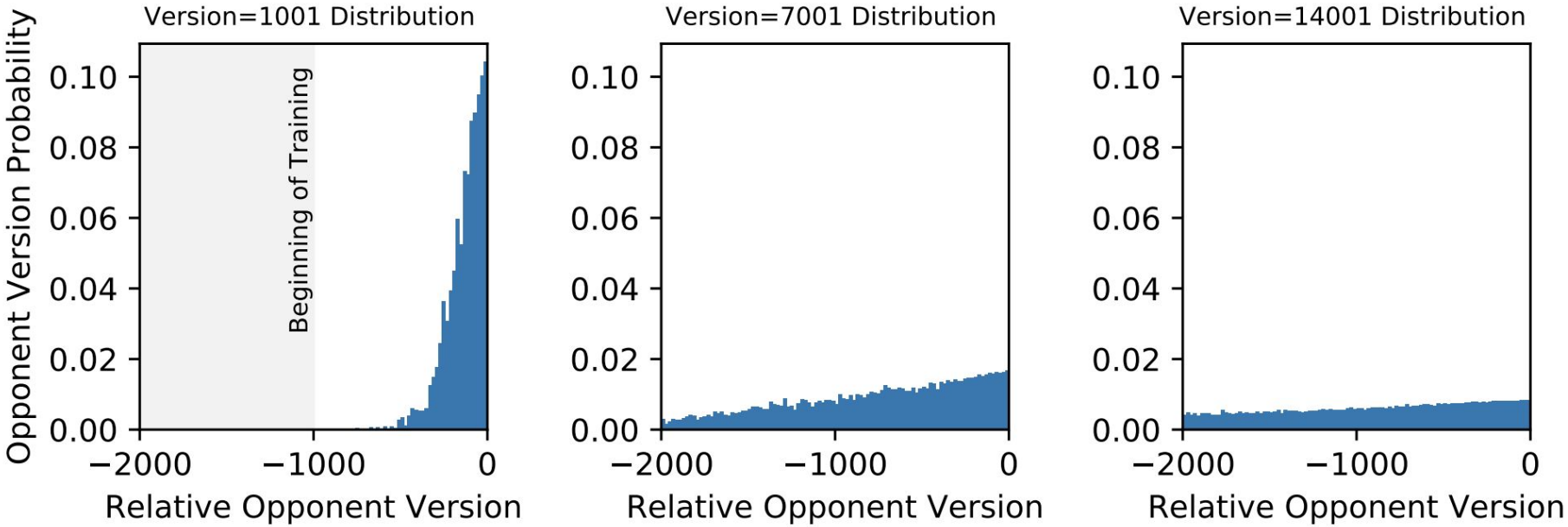
- OpenAI Five was able to beat the international champions on April 2019
- Was the first AI to beat world champions at an e-sports game
- Also beat 99.4% of 7,257 games against public challengers
- Some limitations
  - Played with 17 of the game's 117 heroes
  - No support for certain items that create additional controllable units



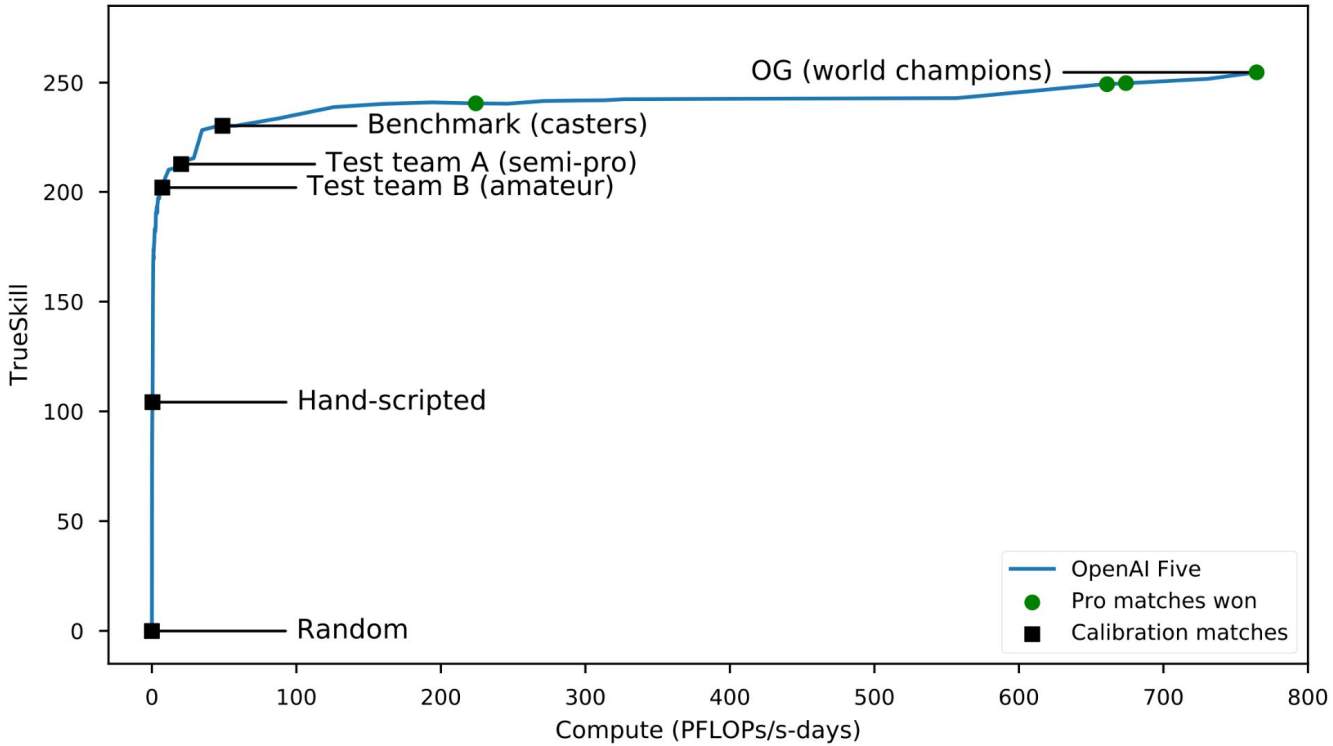
Training process

- OpenAI Five was trained without any human gameplay data through self-play
- 80% of the games in training are played against the latest set of parameters, and 20% play against past versions
- PSRO-type: past versions are used to obtain more robust strategies and avoid “strategy collapse”
- Dynamic sampling system: each past opponent is given a quality score which is occasionally updated. Opponent agents are sampled according to a softmax distribution

Opponent distribution



Training progress



Key to success

- The key ingredient was to “scale existing reinforcement learning systems to unprecedented levels, utilizing thousands of GPUs over multiple months” (Ten months of training using ~800 PFlops/s·days of compute)
- Deep learning is again crucial: the policy is parameterized by a recurrent neural network, comprising a single-layer LSTM
- Neither OpenAI Five nor AlphaCraft focus on visual processing, but rather on the decision making aspect
    - The model can see all the information available simultaneously, whereas a human needs to actively click to see various parts of the map

AlphaStar vs. OpenAI Five

	AlphaStar	OpenAI Five
Starting point	Imitation learning from human replays	Self-play without gameplay data
Training	League of main agents and exploiters	80% current policy, 20% historical versions
Opponent selection	PFSP to prioritize strong opponents	Adaptive scores prioritize hard opponents
Evaluation	Grandmaster, anonymous matches	Defeated world champion

References

- Brian Hu Zhang, Tuomas Sandholm (IJCAI 2024): “Exponential Lower Bounds on the Double Oracle Algorithm in Zero-Sum Games”
- McAleer et al. (2022): Anytime PSRO for Two-Player Zero-Sum Games
- McAleer et al. (ICLR 2024): Toward Optimal Policy Population Growth in Two-Player Zero-Sum Games
- Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Perolat, David Silver, Thore Graepel (NeurIPS 2017): “A unified game-theoretic approach to multiagent reinforcement learning”
- Vinyals et al. (Nature 2019) “Grandmaster level in StarCraft II using multi-agent reinforcement learning”
- OpenAI (2019): Dota 2 with Large Scale Deep Reinforcement Learning