Introduction to Reinforcement Learning

A mini course @ HCMUS, Vietnam

Lectures 10-12 (cont'd)

Long Tran-Thanh

long.tran-thanh@warwick.ac.uk

University of Warwick, UK

When LLM Agents Strategically Behave

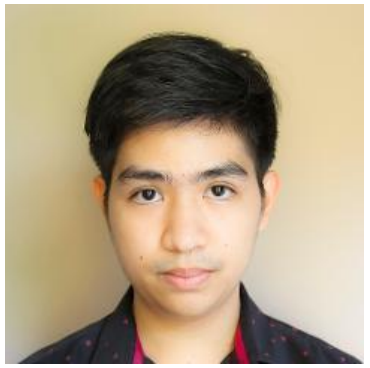
In this talk: ~~NO~~ minimal technical formulae !!!

regret minimising talk
(with some colourful slides)

Sam Taaghol

Hoang Pham

Nam Tran



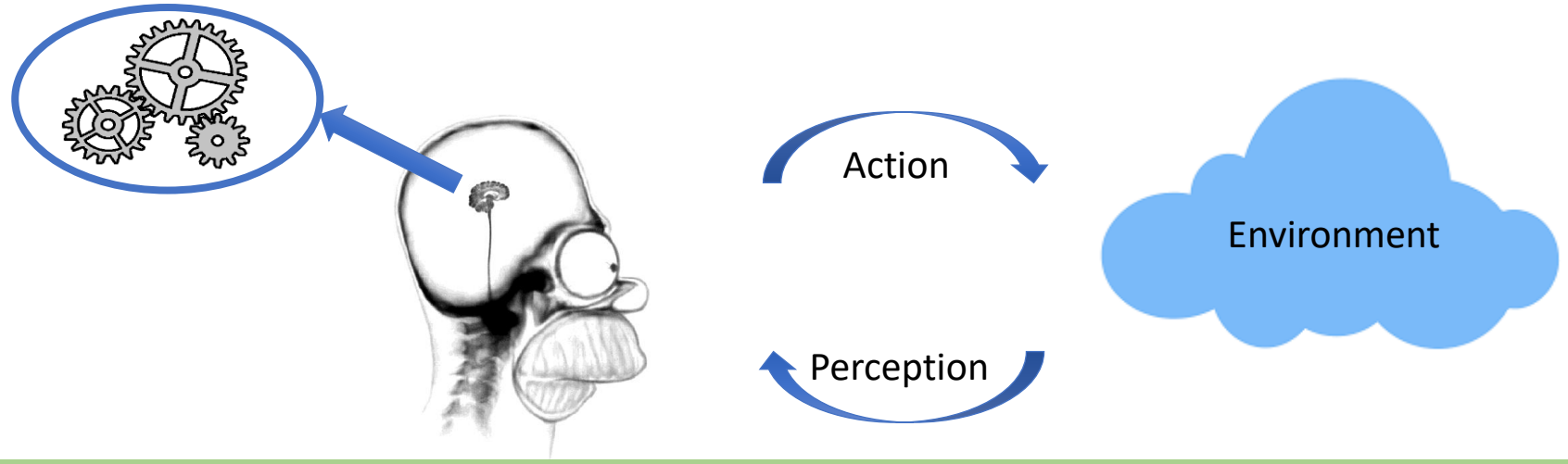
Nut Srisawad

Abhimanyu Pallavi Sudhir

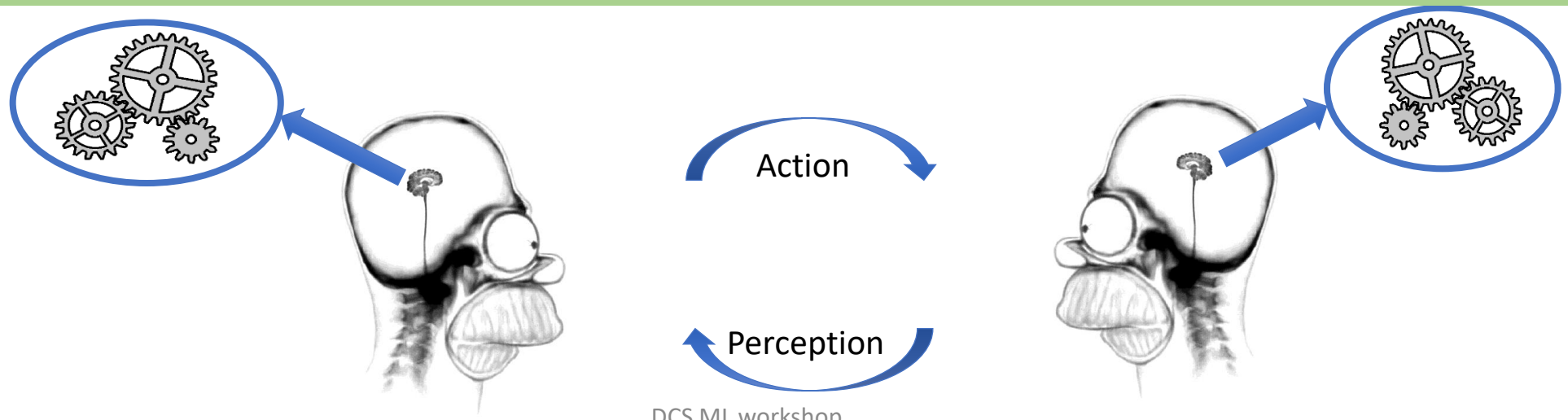
Sharlin Utke

Learning Agent Model

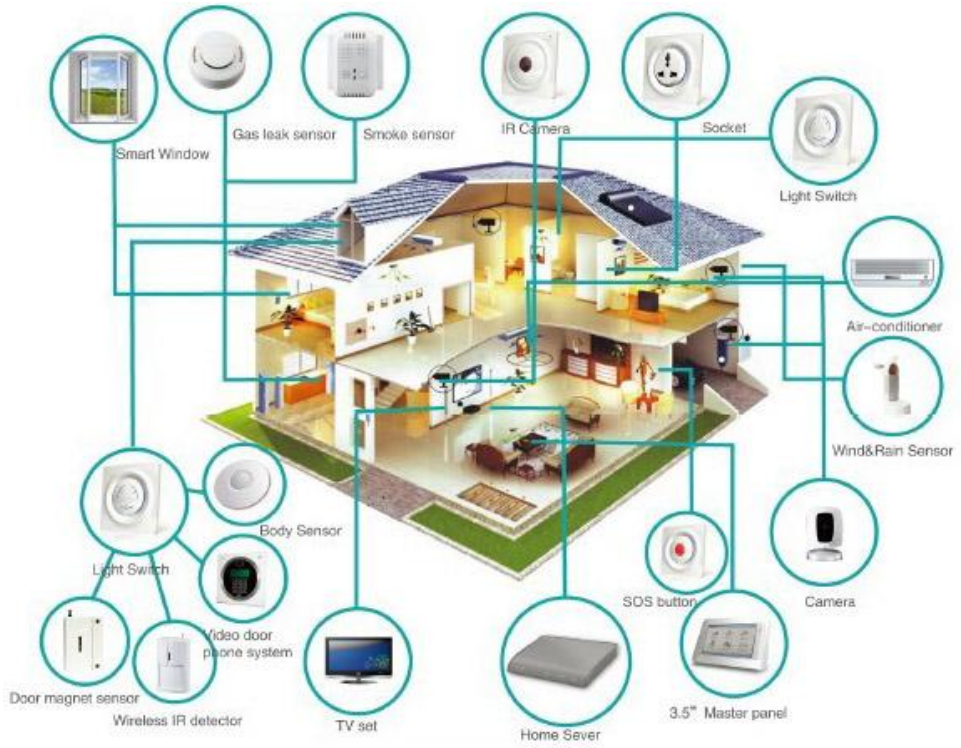
Single-agent setting



Multi-agent setting



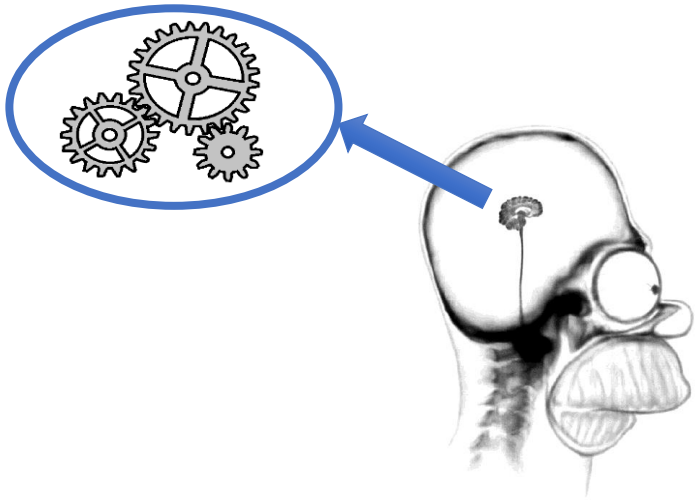
Future of LLM Agents



LLM Agent & Human(s)

Multi LLM Agents (+ mixed humans)

How Strategic Agents Behave?



Definition 1: Strategic agent = utility maximiser

Definition 2: Strategic agent = regret minimiser

Typical concept: no-regret learner (~ average regret converges to 0 over time)

Topics covered today

Behaviour of no-regret learners is easy to predict

- Application : Last-round/last-iterate convergence in multi-agent learning

It’s easy to attack no-regret learners

- Attack against bandits : attack upper and lower bounds
- Attack against RL agents: the necessity of multimodal attacks

It’s easy to fool no-regret learners

- Learning against deceivers and how to counteract
- Learning in coopetitive games

Topics NOT covered

Learning with structured data - PhD student: Nam Tran

- Symmetric bandits: strict generalisation of sparsity (ECML'23, NeurIPS'24)
- Learning with geometric structures (NeurIPS'24)

Deep learning:

- Pruning at initialisation – PhD student: Hoang Pham (NeurIPS'23, ICLR'25)
- Lifelong deep learning (CVPR'22)

Robotics:

- Human-robot collaboration – PhD student: Balint Gucsi at Southampton (IROS'20, HRI'25, ICRA'25)

AI for Social Good:

- E.g, wildfire mitigation (IJCAI'20), vaccine allocation (IJCAI'22), homeless housing program (AIES'18), antipoaching (IJCAI'19), etc.

1. Strategic learners are easy to predict

Joint work with Le Cong Dinh, Tri-Dung Nguyen, Alain Zemkoho (Southampton)

(ALT 2021, JAAMAS 2023)

Problem setting

Repeated 2-player zero-sum game: agent, adversary

- At each t = {1,..T}: agent chooses strategy (action) $f_t \in \mathcal{F} \subseteq [0,1]^n$
- Adversary simultaneously chooses strategy $x_t \in \mathcal{X} \subseteq [0,1]^n$
- Agent observes loss $\langle f_t, x_t \rangle$ and x_t (full information feedback)
- Adversary is a no-regret learner:

$$\frac{1}{T} \max_{x \in \mathcal{X}} \sum_{t=1}^{T} (\langle f_t, x \rangle - \langle f_t, x_t \rangle) \to 0, \quad T \to \infty$$

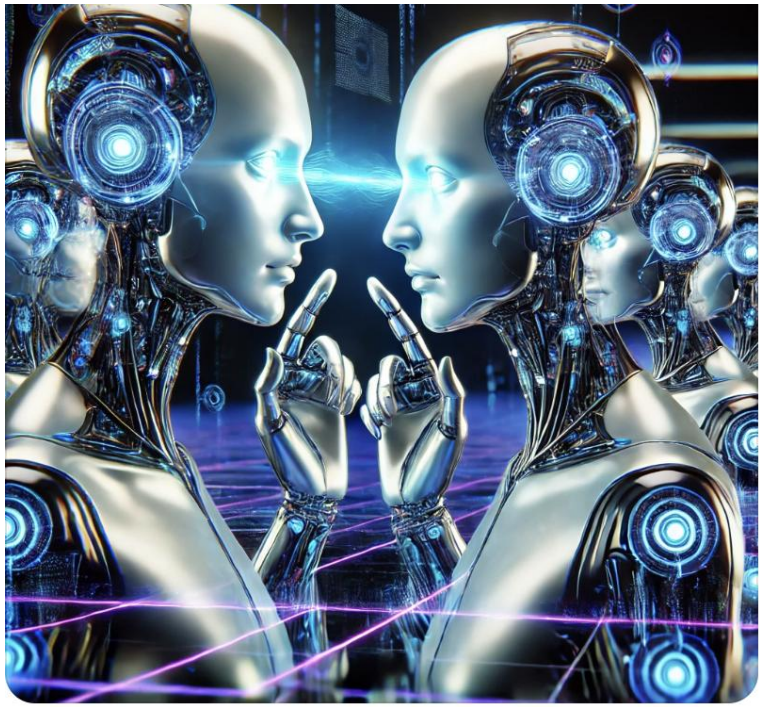
No-regret learners are easy to predict

If adversary uses no-regret learning algorithm to choose x_t at each time step *t* -> Can we predict the next step of adversary without the need to observe it?

Idea: design a special algorithm (Accurate Follow the Regularized Leader) and play against the adversary

Key result: We show that $||x_{t+1} - x_t||_q \in O\left(\frac{1}{\sqrt{T}}\right)$ -> use x_t to predict x_{t+1} (we assume we observe adversary's move **after** we have made our own)

Application: last-round convergence in multi-agent systems



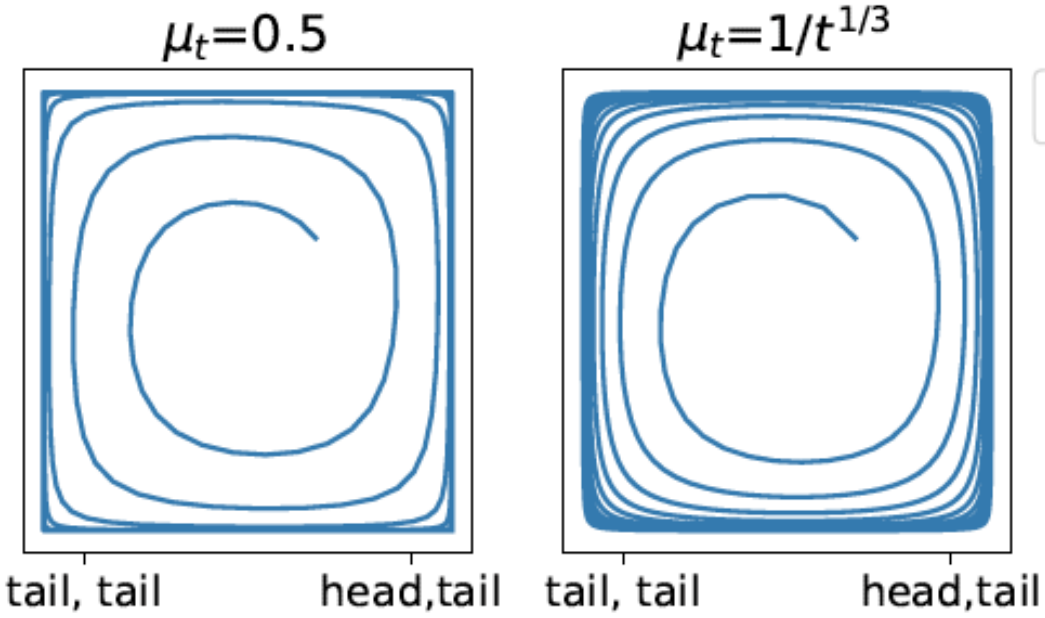
Selfish behaviour causes system collapse:

- Piatti *et al.*, Arxiv'24 (e.g., financial system's collapse)

Impossibility results

Repeated Matching Pennies after 2500 iterations:

- No-regret learning alg: Multiplicative Weight Update
- Blue line: MWU vs MWU
- System dynamics: outward spiral -> no convergence



Known since Mertikopoulos, Papadimitriou & Piliouras (2018): **no last round convergence** in general case. Other notable work: Bailey and Piliouras (2018), Cheung and Piliouras (2019)

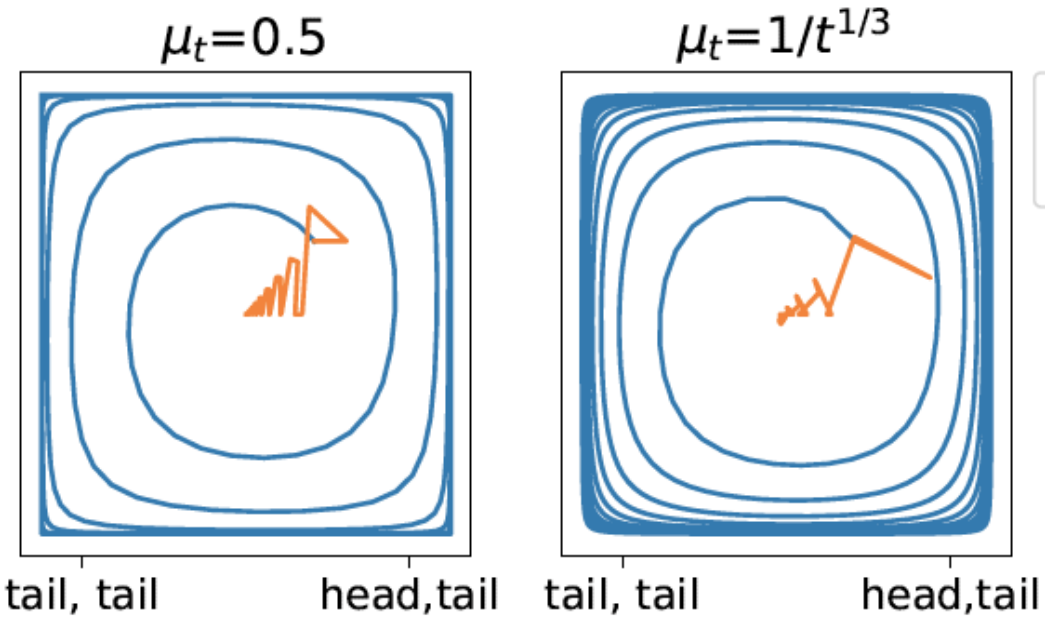
Existence of last round convergence – some special cases:

- Daskalakis and Panageas (2018): Optimistic MWU + unique minimax equilibrium
- Bu, Ratliff & Mesbahi (2019): Differential games (linear-quadratic) + gradient ascent/descent
- Goktas & Greenwald (2022): Exploitability-minimising strategy profiles

Last-round convergence with asymmetric knowledge (Dinh *et al.*, ALT 2021)

Asymmetric information:

- Leader (column player) can estimate her (approximate) minimax strategy
- Follower (row player) is a regret minimiser -> uses a no-regret algorithm
- Leader's objectives:
  - Last round convergence
  - Provable performance guarantee (e.g., no-regret)



Main result: Column player uses LRCA and row player chooses from a large class of no-regret algorithms (including FTRL-class) -> **last round convergence** + column player achieves **no-instant-regret**

2. It’s easy to attack no-regret learners

Joint work with and Sam Taaghol (Warwick), Shivakumar Mahesh (Warwick/Oxford), Anshuka Rangi (UCSD/Amazon), Haifeng Xu (UChicago) and Massimo Franceschetti (UCSD)

(AAAI’22, IJCAI’22, ICML’25 submission)

Exploration vs. exploitation

Trade-off

Exploration = learning the model

(learn the reward values)

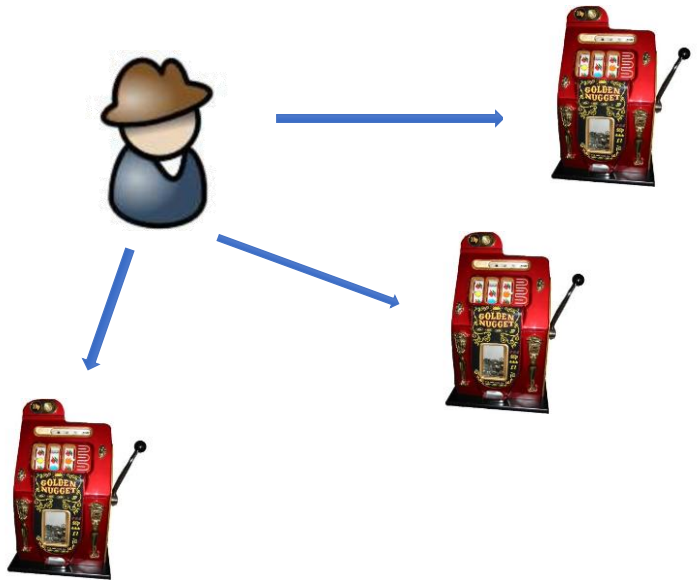
Exploitation = optimising over learnt model

(optimise the rewards over time)

Too much exploitation: not enough information -> suboptimality

Too much exploration: not enough time to optimise -> suboptimality

The multi-armed bandit (MAB) model



There are multiple arms

At each time step (round):

- We choose 1 arm to pull
- Receive a reward, drawn from an unknown distribution of that arm

Objective: maximise the expected total reward

Exploration: we want to learn each arm's expected reward value

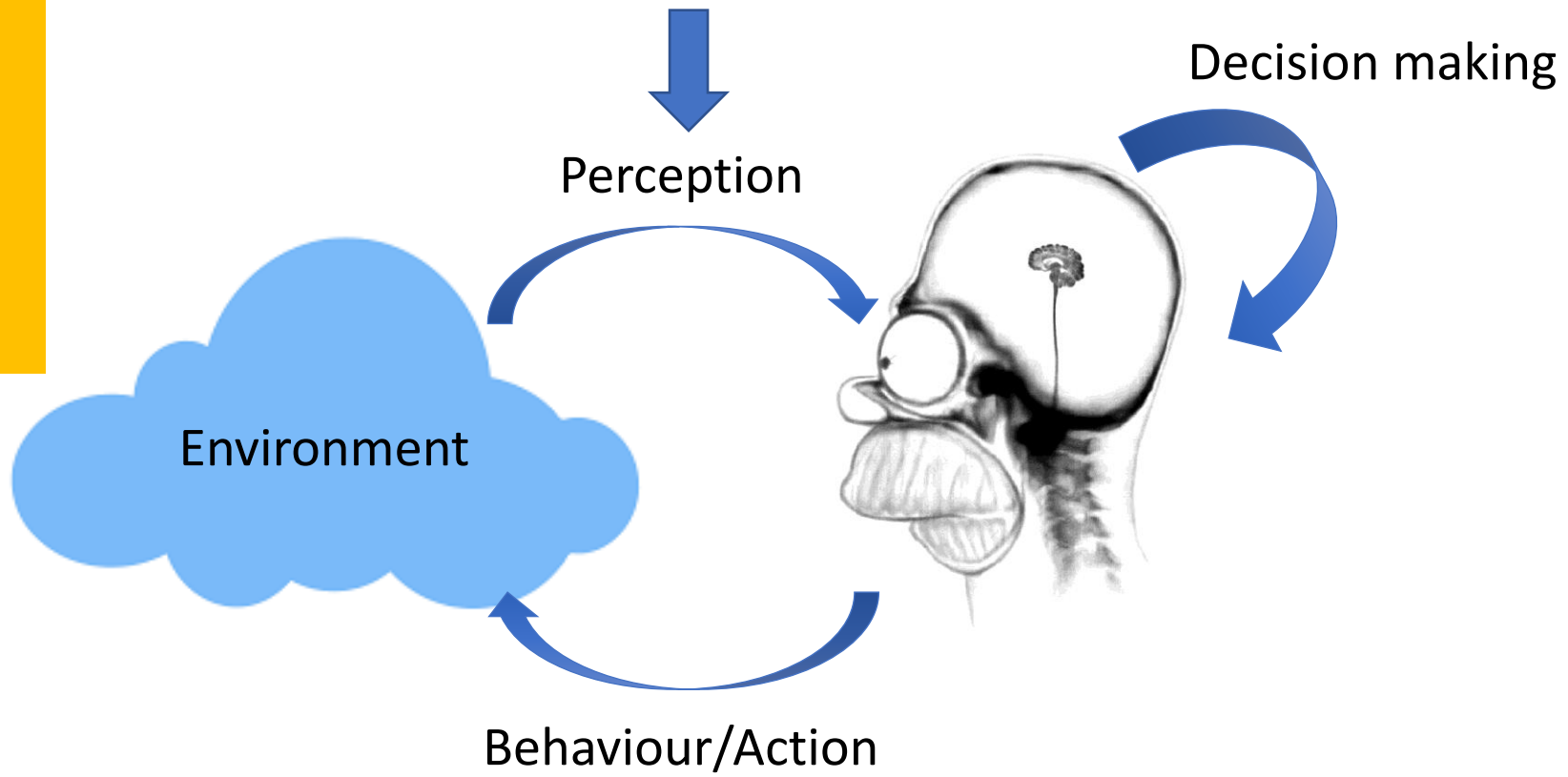
Exploitation: we want to maximise the sum of the rewards

MAB is the **simplest (and flexible) model** that captures the dilemma of exploration vs. exploitation

Attack on bandits

Attacker objective:

- **To induce linear regret** (i.e., to violate the no-regret property)



Characterisation of attackers

- Strength of attackers
  - Weak attacker: makes contamination each time step **before** observing the pulled action
  - Strong attacker: makes contamination each time step **after** observing the pulled action

- Contamination budget bounds
  - Bounded above surely by a threshold (deterministic budget)
  - Bounded above in expectation (expected budget)

How easy to attack the bandit models?

Assumption: bandit algorithm is **oblivious to the attack**

Question 1: How much contamination is needed to succeed (i.e., to induce linear regret)?

Answer: typically O(logT) attacks is sufficient to succeed

Question 2: What is the minimum contamination needed to succeed?

Existing work: no lower bound. Conjecture: strong attacker needs less than weak one

Our work:

- **Weak = strong attacker** (in most cases)
- **First attack lower bounds** (using notion of conservativeness)

Attacking Episodic RL agents

Episodic RL = bandit with state transitions

Multimodality of attacks: adversary can contaminate reward, state transition, action (or combination of these)

Result 1 (Rangi *et al.,* IJCAI'22): Single mode attacks are not feasible in bounded reward settings (i.e., rewards are between 0 and 1)

Result 2 (Rangi *et al.,* IJCAI'22): Reward + action manipulation needs $\tilde{\Theta}(\sqrt{T})$ contaminations

Defence against these attacks

Question 1: how to design attack-aware bandit algorithms to mitigate the attack

Setting: Deterministic contamination budget C
(no results on expected budget to date)

	Strong attacker	Weak attacker
Regret upper bound	$O(C^2 + C\log T + \sqrt{T})$ (Bogunovic et al. 2020)	$O(KC + (\log T)^2)$ (Gupta, Koren, and Talwar 2019)
Regret lower bound	$\Omega(C)$	$\Omega(C)$

Question 2: Can we go below the lower bound of C?

Saving bandits with verification: unlimited version

Agent's objective: Minimise the number of verifications required to restore the logarithmic regret bound.

Theorem 2: Lower Bound (Rangi et al., AAAI'22)

The minimum number of verifications required to restore logarithm regret is at least $\Omega(\log T / min_{i_1, i_2 \in [K]} KL(i_1, i_2))$

Remark: This lower bound is tight and can be achieved by multiple verification algorithms

Saving bandits with verification: limited version

Agent's objective: Minimise the regret if the number of verifications are bounded by at most *B.*

Motivation: If *B=0,* then the regret scales at least $\Omega(C)$, where *C* is the amount of contamination.

Question: If *B>0,* then can we do better then this lower bound of $\Omega(C)$?

Answer: Yes.

Regret upper bound for verification with limited budget

Theorem 3: Upper Bound (Rangi et al., AAAI'22)

With probability at least $1-\delta-\beta$, the regret of Secure-BARBAR against any weak attackers with amount of contamination at most C is bounded by

$$O\left(K \min\left\{C, \frac{T \log \frac{2}{\beta} \ln(\frac{8K}{\delta} \log_2 T)}{\sqrt{B/K}}\right\} + \sum_{i \neq i^*} \frac{\log T}{\Delta_i} \log\left(\frac{K}{\delta} \log T\right)\right).$$

Remark: This is a tight upper bound (i.e.., there's a matching lower bound)

3. Learning against strategic manipulators (i.e., deceivers)

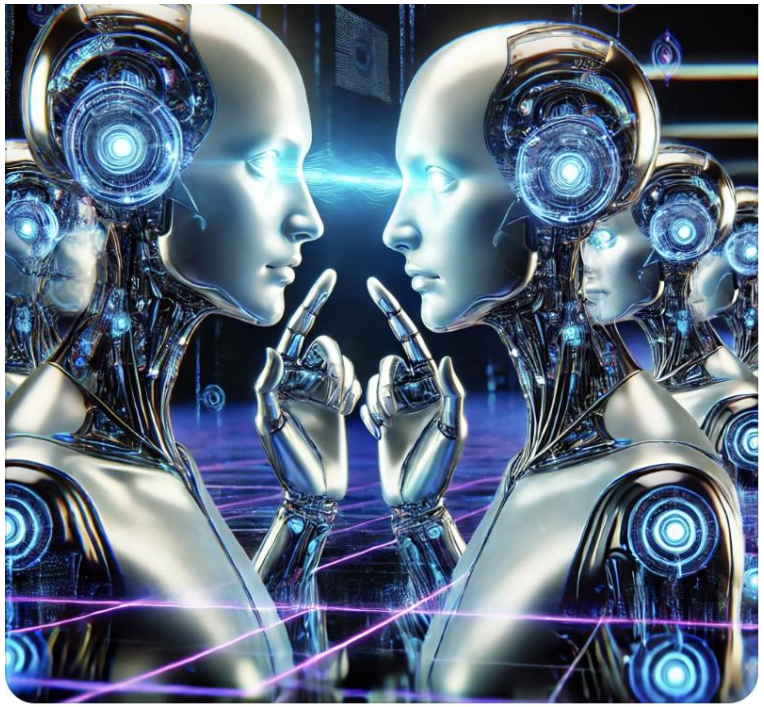
Joint work with Jiarui Gan, Nick Bishop (Oxford)

Qingyu Guo, Bo An (NTU)

Enrico Gerding (Southampton)

(NeurIPS'19, ACM EC'19, NeurIPS'20)

Motivation



LLM agents can exploit each other:

- Deceive their human users/fellow agents (e.g., to steal their money) (Scheurer *et al.*, ICLR workshop'24)

Learning with strategic manipulators

2-player general-sum game

Setting description:

- Manipulator aims to maximise their (unknown) utility function
- Manipulator can modify the observation of the learner (data, payoff, etc)

Mimicking zero-sum game is the best for the manipulator (Gan *et al.*, NeurIPS 2019)

- 2-player non-zero-sum **security games**
- Opponent **can lie about their type** (e.g., type = payoff parameters) to fool us, **but we can also see the reported payoff** of the opponent
- Question: what manipulation would benefit the opponent the best

Brain teaser: which one is better?

Playing zero-sum game, but pretend that you are not vs.

Playing non-zero sum game, but pretend that you are full adversary (i.e., play a zero-sum game)

Theorem (informal): The best strategy for the manipulator is to **pretend to play a zero-sum game** (i.e., we are back to the minimax optimisation problem in the worst-case)

Counteracting this manipulator (Gan *et al.*, NeurIPS 2019)

- The previous result holds if the defender (a.k.a. we) are oblivious to the manipulation
- If we are aware to the fact of manipulation: **We can do better than solving the minimax problem**

Policy: Set of rules that determines what to play against each reported type (aka payoff matrix)

Stackelberg (a.k.a. leader-follower) model: We first announce our policy, then the opponent makes the best response (i.e., compute the best lie)

Main result: The optimal policy can be calculated in **polynomial time**, and the resulted solution is **strictly better** than that of the minimax problem in many cases.

4. Coopetitive setting

Joint work with Shivakumar Mahesh (Warwick/Oxford), Nick Bishop (Oxford), Le Cong Dinh (Southampton)

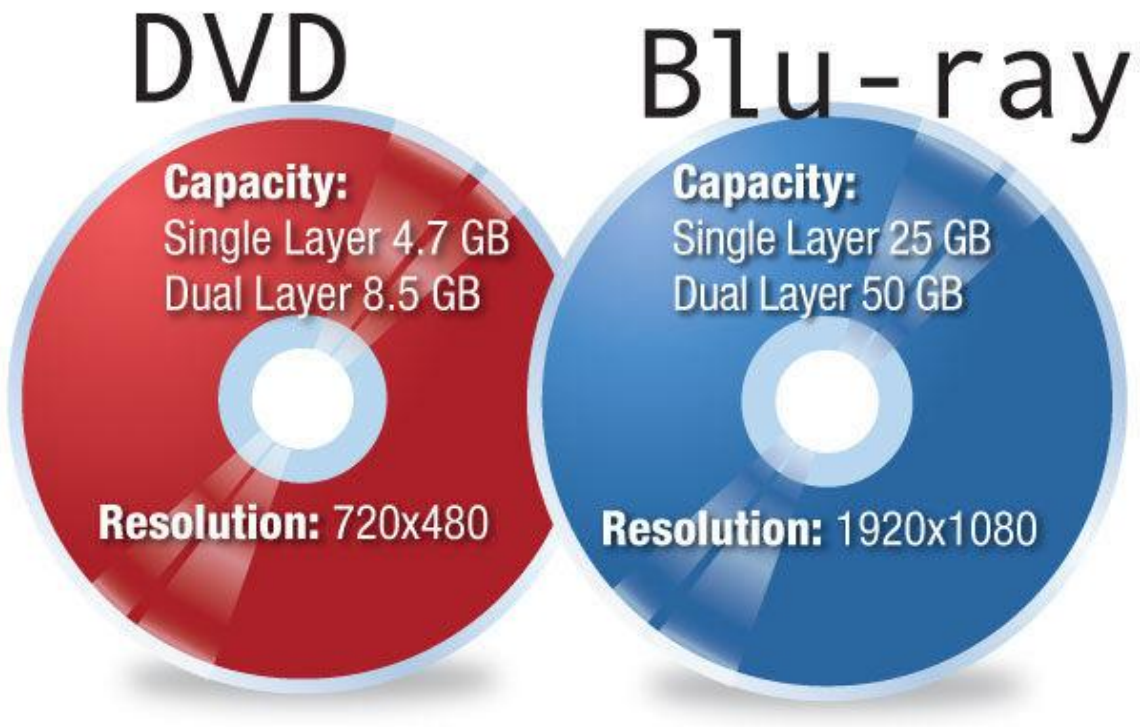
(GameSec'23, Arxiv)

What’s coopetitive game?

- In order to win/perform well, one must cooperate with their opponents
- But they also need to know when to stop cooperating to become the winner/achieve their goal
- That is, they need to **cooperate and compete** at the same time (Nalebuff & Brandenburger, 1996)

https://cruciformstuff.com/2023/07/30/betrayal/

Example 1: Blue-Ray vs. DVD



https://fr.tipard.com/resource/blu-ray-vs-dvd.html

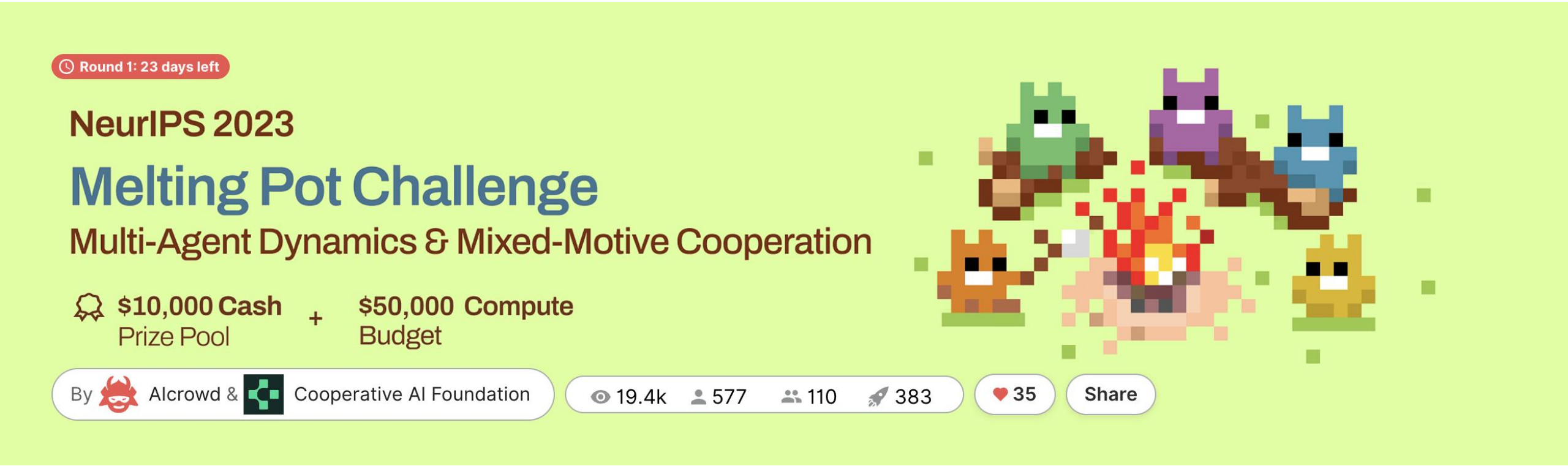
Example 2: Tour de France

https://www.ef.fr/blog/language/les-principaux-termes-de-cyclisme-connaitre-pour-regarder-le-tour-de-france/

Recent interests from the AI Community

Google Deepmind + Cooperative AI Foundation's Melting Pot Challenge (hosted at NeurIPS 2023)

https://www.aicrowd.com/challenges/meltingpot-challenge-2023



Research questions

In AI, we consider a multi-agent sequential decision-making version of **coopetitive games**:

- Who to cooperate with?
- How to signal/incentivise others to collaborate
- When to switch side?

Our focus

- Aim: Proof of Concept
- Simplified setting
- 3 players
- Repeated games
- Polymatrix games
- Signaling: payoff manipulation

Payoff manipulation explained

- In our setting no explicit communication between agents is allowed
- Instead, we allow one agent to modify another agent's payoff by:

  - Sacrificing from their own payoffs (e.g., gift, bribery, etc) -> increasing the other's payoff
  - Enforce some penalties -> decreasing opponent's payoff
  - Examples: multiplayer video games, nature, etc.

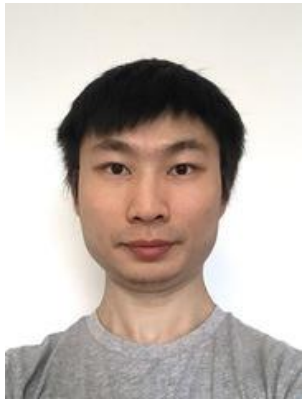
Main Results (Mahesh *et al.*, GameSec 2023)

(Informal) We prove that the **manipulating agent can learn to manipulate with minimal cost** to win the games

Applied to iterated prisoners' dilemma: can manipulate opponents to cooperate, while we can still defect them

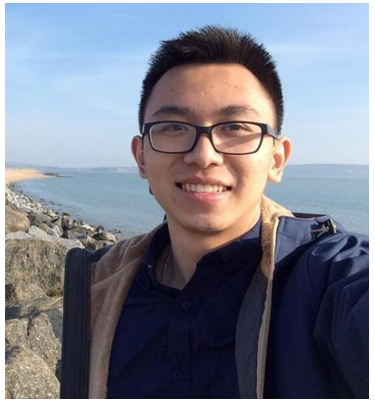
And many more results: https://arxiv.org/abs/2110.13532

Many thanks for your attention



Jiarui Gan

Nick Bishop



Le Cong Dinh

Shiva Mahesh

Anshuka Rangi

Bo An

Mike Wooldridge

Massimo Franceschetti