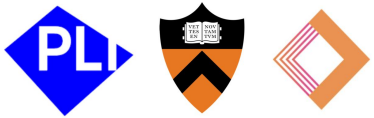


Lecture: Multi-Agent RL

Jiayi Geng
PLI, Princeton NLP, AI Lab
jiayig@princeton.edu
2025/04/17

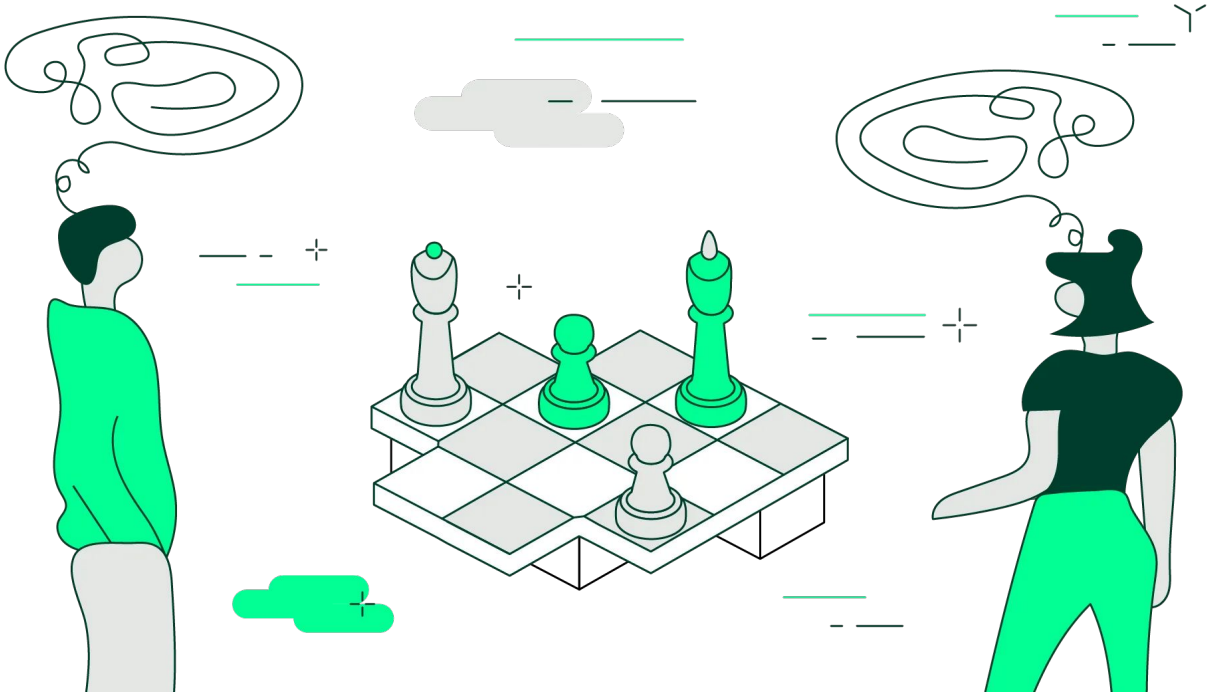
Overview

- Multi-agent RL:
  - Game theory
  - Multi-agent learning
- Multi-agent meets large language model:
  - Multi-agent communication
  - Multi-agent collaboration

Overview

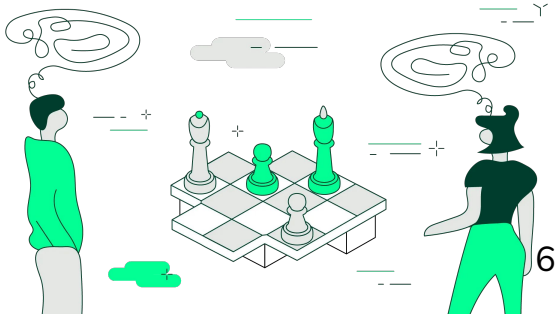
- **Multi-agent RL:**
  - **Game theory**
  - **Multi-agent learning**
- Multi-agent meets large language model:
  - Multi-agent communication
  - Multi-agent collaboration

Basic Concepts of Game Theory



Three fundamental components of Game Theory

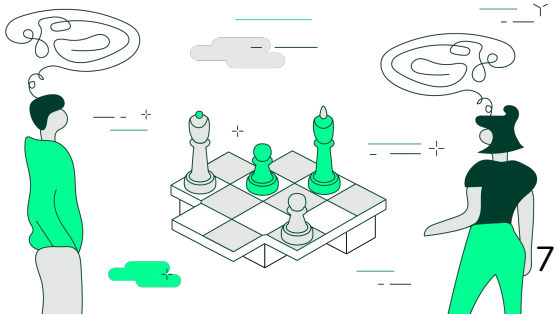
Players: The decision-makers in the game. Players can be individuals or entities.



Three fundamental components of Game Theory

Players: The decision-makers in the game. Players can be individuals or entities.

Strategies: The actions available to the players, from which they choose their course of action.

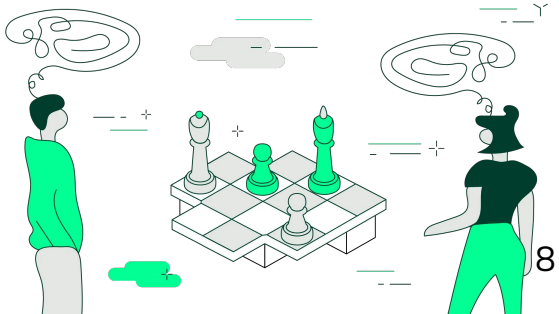


Three fundamental components of Game Theory

Players: The decision-makers in the game. Players can be individuals or entities.

Strategies: The actions available to the players, from which they choose their course of action.

Payoffs: The outcomes resulting from the combination of strategies chosen by all players, usually quantified in terms of utility or rewards.



Nash Equilibrium

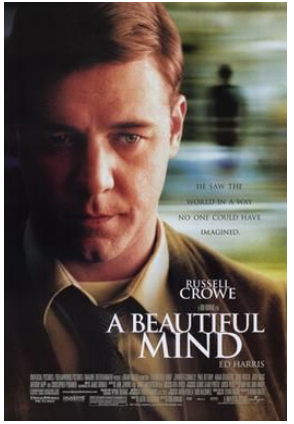
1928-2015

The concept of Nash Equilibrium, named after mathematician John Nash, is central to game theory. It is a situation in which no player can benefit by changing their strategy while the other players keep theirs unchanged.

A strategy profile $s^* = (s_1^*, s_2^*, \ldots, s_n^*)$ is a Nash Equilibrium if for every player i,

$$u_i\left(s_i^*, s_{-i}^*\right) \geq u_i\left(s_i, s_{-i}^*\right)$$

for all s_i in the strategy space of player i, where u_i is the payoff function for player i and s_{-i}^* denotes the strategy profile of all players except i.



Two-Player Zero-Sum Matrix Game

Let’s consider a simple 2x2 matrix:

$$A = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Two-Player Zero-Sum Matrix Game

Let's consider a simple 2x2 matrix:

$$A = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Actions:

- Player 1 chooses the first action
- Player 2 also chooses the first action

Two-Player Zero-Sum Matrix Game

Let's consider a simple 2x2 matrix:

$$A = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Actions:

- Player 1 chooses the first action -> Player 1 gains 1
- Player 2 also chooses the first action -> Player 2 loses 1

Maximin and Minimax Strategies

- **Maximin Strategy:** Player 1 tries to maximize the minimum payoff they can guarantee:

$$\max_{\mathbf{x}} \min_{\mathbf{y}} \mathbf{x}^T A \mathbf{y}.$$

- **Minimax Strategy:** Player 2 aims to minimize their maximum possible loss (equivalently, minimizing Player 1's maximum payoff):

$$\min_{\mathbf{y}} \max_{\mathbf{x}} \mathbf{x}^T A \mathbf{y}.$$

Saddle Point & Equality

Strong duality in linear programming and convex optimization ensures that for a zero-sum game,

$$\max_{\mathbf{x}} \min_{\mathbf{y}} \mathbf{x}^T A\mathbf{y} = \min_{\mathbf{y}} \max_{\mathbf{x}} \mathbf{x}^T A\mathbf{y},$$

implying the existence of a saddle point where both strategies are optimal and no unilateral change is beneficial.

Learning in Games

How to solve for Nash: Best response?

Best Responses Dynamics

Best response dynamics is a process in which players iteratively adjust their strategies to best respond to the strategies currently played by their opponents

Best Responses Dynamics

Best response dynamics is a process in which players iteratively adjust their strategies to best respond to the strategies currently played by their opponents

Suppose at time t, Player 1 and Player 2 updates their strategies are given by:

$$\mathbf{x}_{t+1} = \mathrm{BR}_1(\mathbf{y}_t), \quad \mathbf{y}_{t+1} = \mathrm{BR}_2(\mathbf{x}_t),$$

Fictitious Play

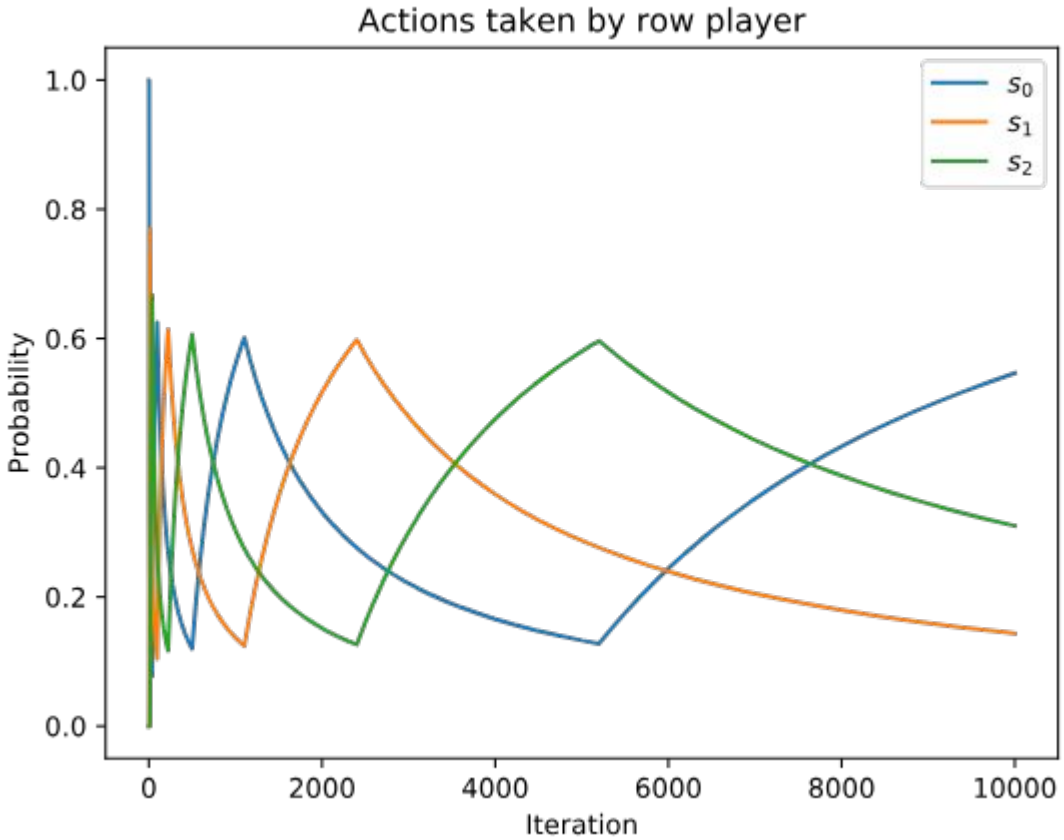
Fictitious play is a simple iterative learning process in which each player assumes that their opponents are playing fixed and mixed strategies.

Fictitious Play

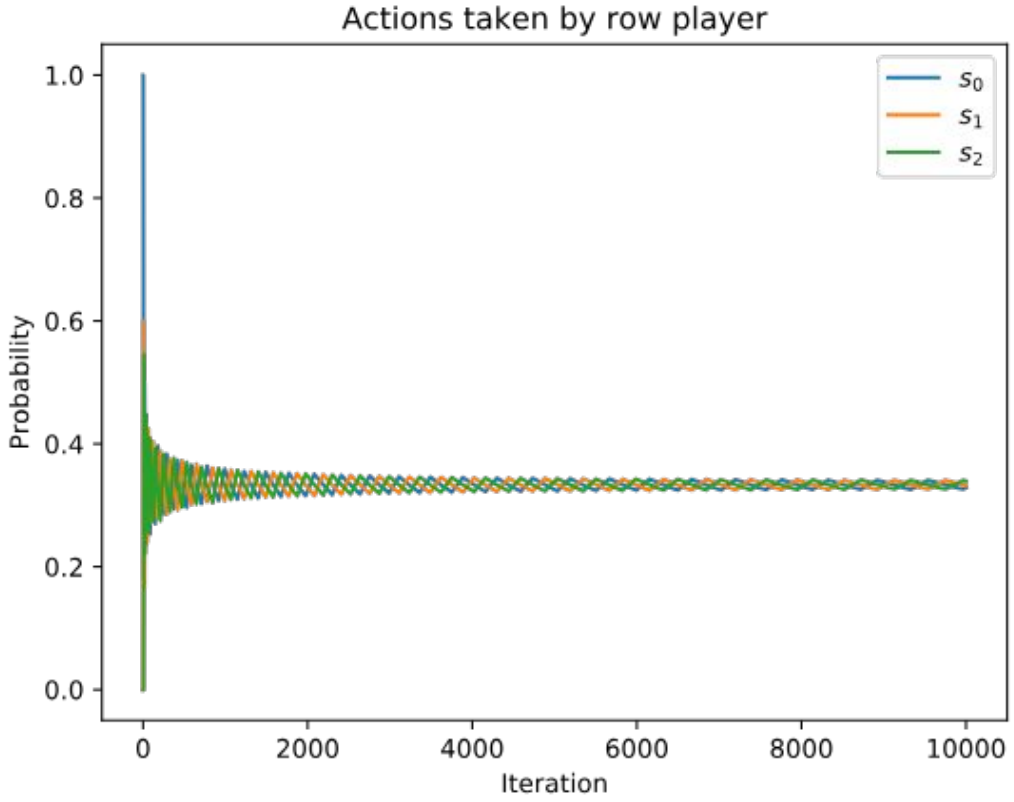
Fictitious play is a simple iterative learning process in which each player assumes that their opponents are playing fixed mixed strategies.

“Track the opponent’s past actions and best-respond to their frequency."

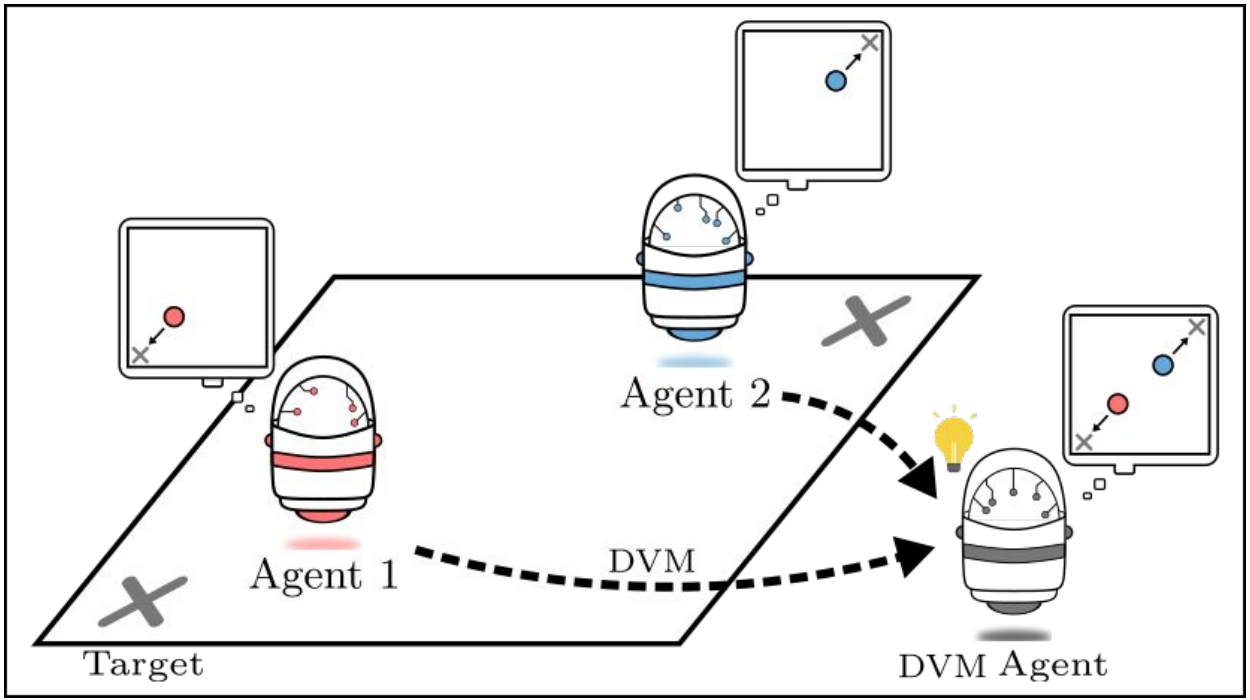
Convergence of Fictitious Play



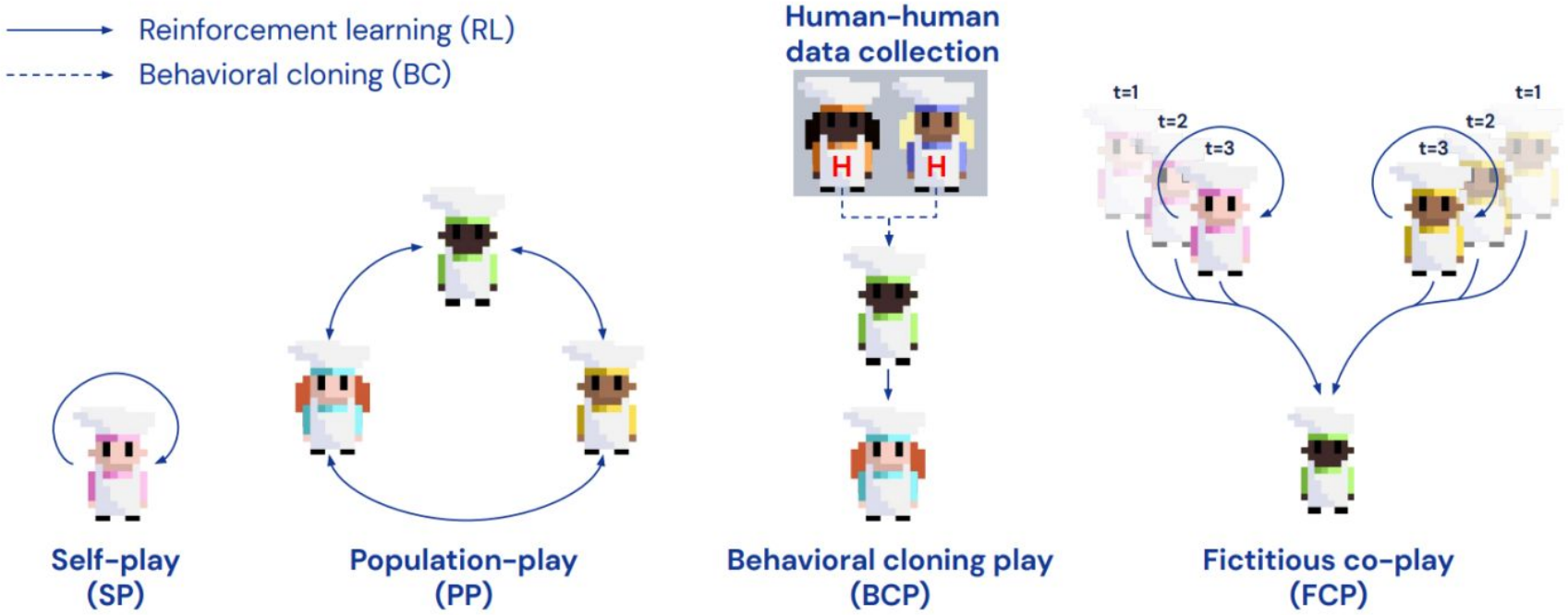
Convergence of Fictitious Play



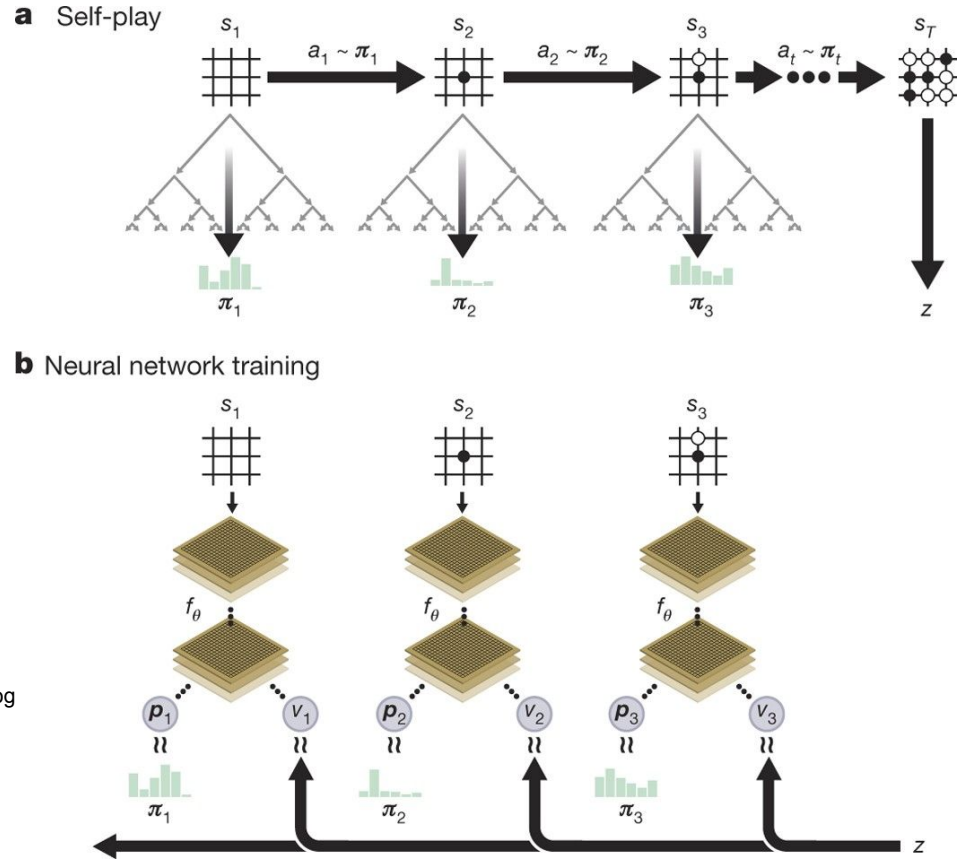
Multi-Agent Learning



Key idea of multi-agent RL: Self-Play



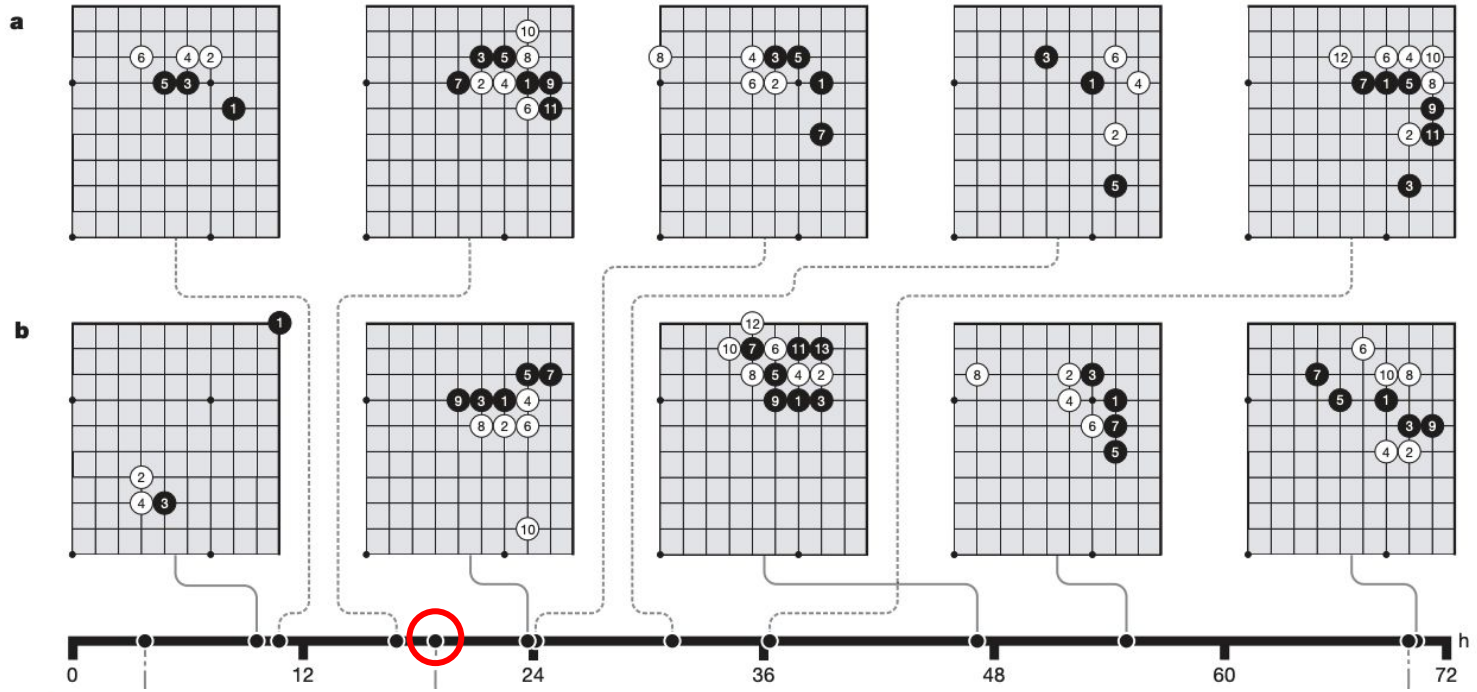
Self-play in Multi-Agent RL: AlphaGo Zero (2017)



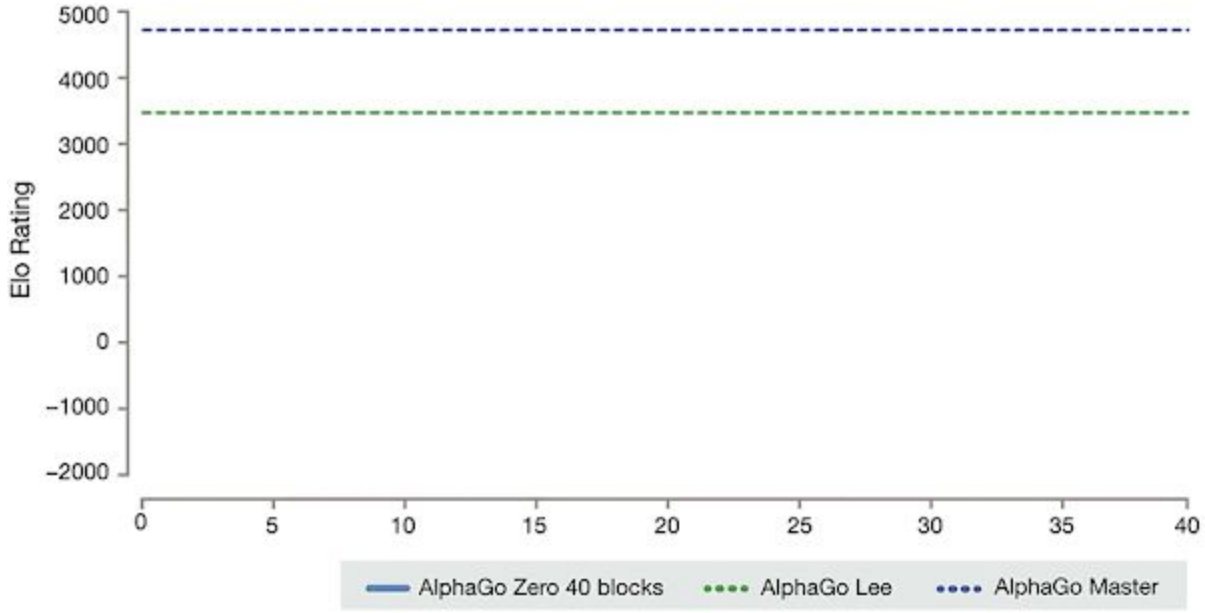
Source:https://deepmind.google/discover/blog
hago-zero-starting-from-scratch/

Self-play in Multi-Agent RL: AlphaGo Zero (2017)

RL evolves over time, finding new strategies and phase out old ones (rapid reranking).



Self-play in Multi-Agent RL: AlphaGo Zero (2017)



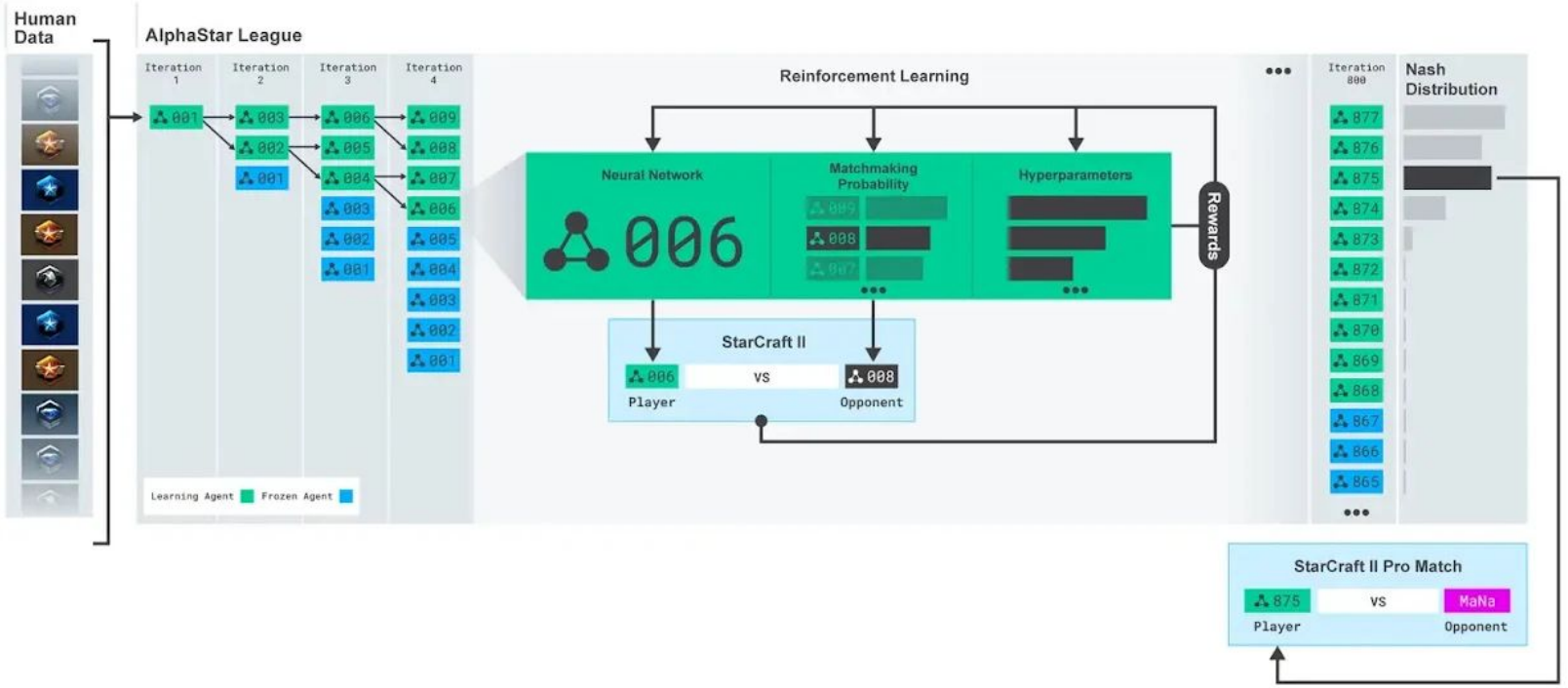
Source:https://deepmind.google/discover/blog/alphago-zero-starting-from-scratch/

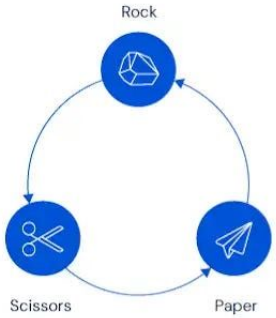
Self-play in Multi-Agent RL: OpenAI Five (2018)

	OpenAI 1v1 bot	OpenAI Five
CPUs	60,000 CPU cores on Azure	128,000 preemptible CPU cores on GCP
GPUs	256 K80 GPUs on Azure	256 P100 GPUs on GCP
Experience collected	~300 years per day	~180 years per day (~900 years per day counting each hero separately)
Size of observation	~3.3 kB	~36.8 kB
Observations per second of gameplay	10	7.5
Batch size	8,388,608 observations	1,048,576 observations
Batches per minute	~20	~60

Deepmind’s AlphaStar (2019)

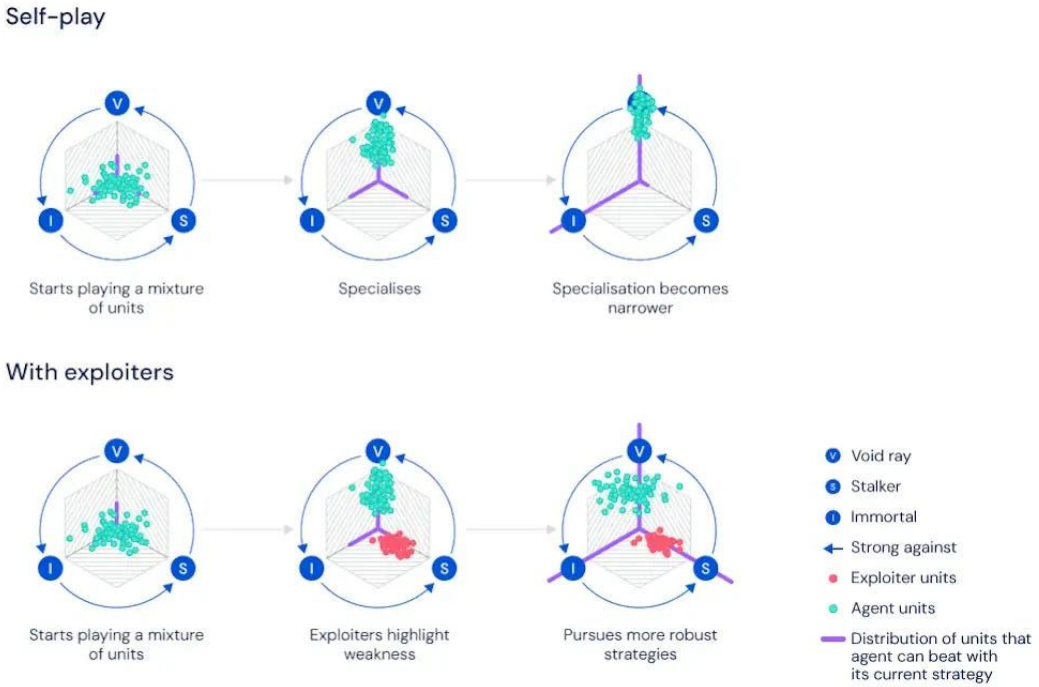
Self-fictitious play in AlphaStar



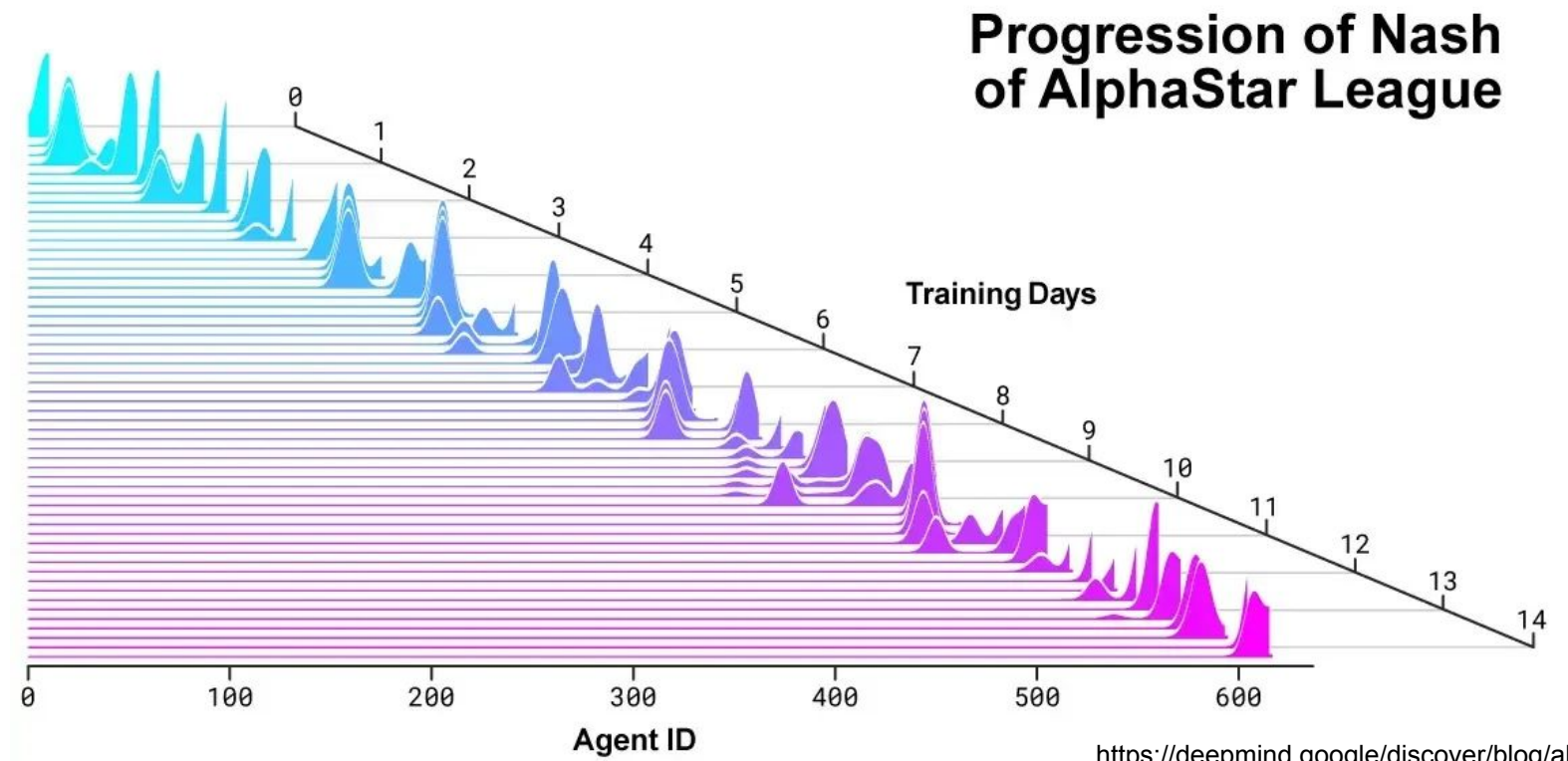


Rock, paper, scissors

StarCraft II players can create a variety of 'units', which have balanced strengths and weaknesses, similar to the game rock, paper, scissors



https://deepmind.google/discover/blog/alphastar-mastering-the-real-time-strategy-game-starcraft-ii/



https://deepmind.google/discover/blog/alphastar-mastering-the-real-time-strategy-game-starcraft-ii/

Overview

- Multi-agent RL:
  - Game theory
  - Multi-agent learning
- **Multi-agent meets large language model:**
  - **Multi-agent communication**
  - **Multi-agent collaboration**

Recap Howard Chen's Lecture

RL Framework for LLMs: LLMs generate answers token by token, similar to an RL agent taking actions in a state.

Recap Howard Chen's Lecture

RL Framework for LLMs: LLMs generate answers token by token, similar to an RL agent taking actions in a state.

Verifiable Reward: The model's response is evaluated by clear, verifiable outcomes (correct vs. incorrect answers), providing a straightforward reward signal.

Recap Howard Chen's Lecture

RL Framework for LLMs: LLMs generate answers token by token, similar to an RL agent taking actions in a state.

Verifiable Reward: The model's response is evaluated by clear, verifiable outcomes (correct vs. incorrect answers), providing a straightforward reward signal.

Group Relative Policy Optimization (GRPO): GRPO refines the model by adjusting log probabilities based on groups of responses.

Recap Howard Chen's Lecture

RL Framework for LLMs: LLMs generate answers token by token, similar to an RL agent taking actions in a state.

Verifiable Reward: The model's response is evaluated by clear, verifiable outcomes (correct vs. incorrect answers), providing a straightforward reward signal.

Group Relative Policy Optimization (GRPO): GRPO refines the model by adjusting log probabilities based on groups of responses.

DeepSeek R1 and Emergent Behaviors for LLM: DeepSeek R1 applies GRPO directly on a strong pretrained model. A approach has led to emergent behaviors.

Multi-Agent RL meets Large Language Models

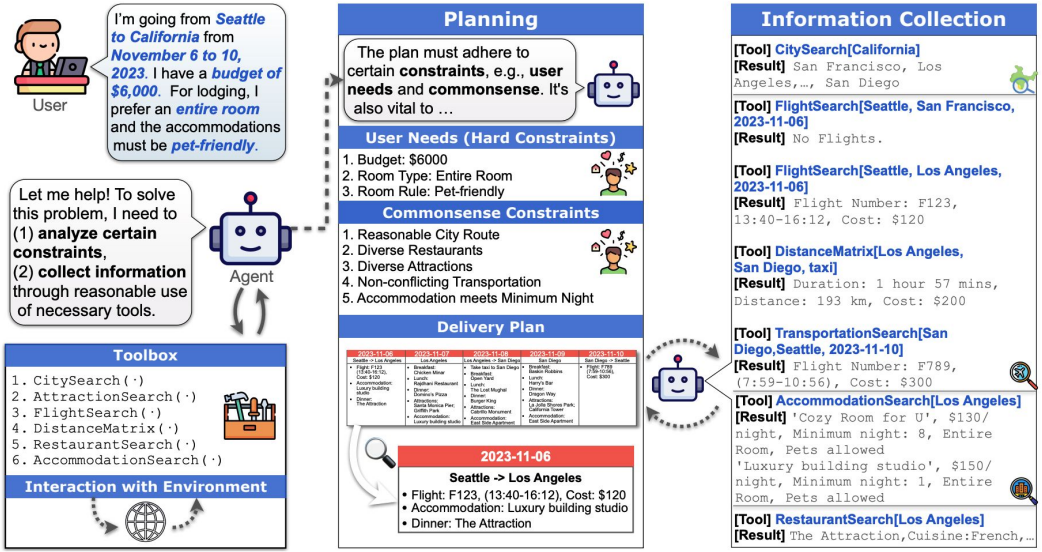
Figure 1: A challenging multi-agent cooperation problem with decentralized control, raw sensory observations, costly communication, and long-horizon multi-objective tasks.

Multi-Agent Communication

LLMs are increasingly capable, why do we need multiple agents and why multi-agent communication is important?

Multi-Agent Communication

LLMs are increasingly capable, why do we need multiple agents and why multi-agent communication is important?



Multi-Agent Communication

LLMs are increasingly capable, why do we need multiple agents and why multi-agent communication is important?

AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation

Qingyun Wu[†], Gagan Bansal[*], Jieyu Zhang[±], Yiran Wu[†], Beibin Li[*]

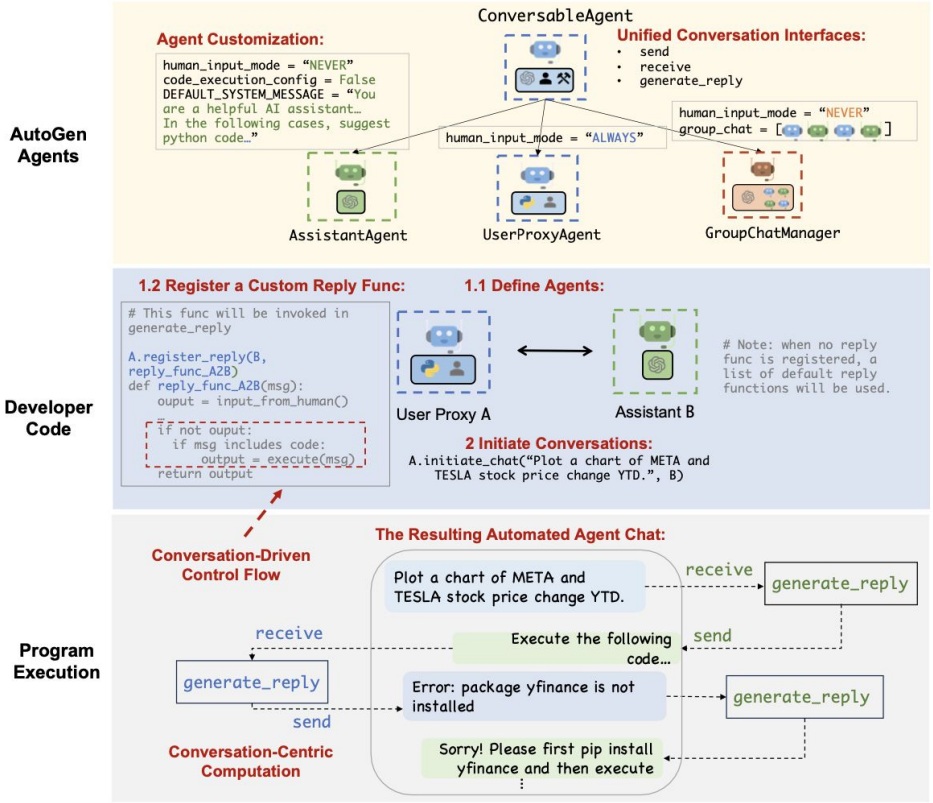
Erkang Zhu[*], Li Jiang[*], Xiaoyun Zhang[*], Shaokun Zhang[†], Jiale Liu[∓]

Ahmed Awadallah[*], Ryen W. White[*], Doug Burger[*], Chi Wang[*1]

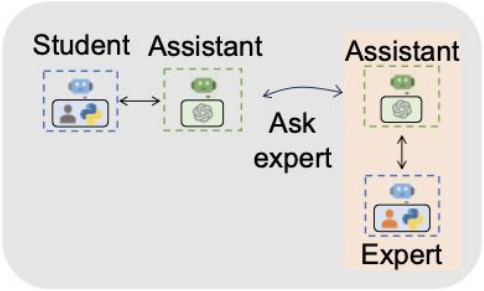
[*]Microsoft Research, [†]Pennsylvania State University

[±]University of Washington, [∓]Xidian University

Core Design

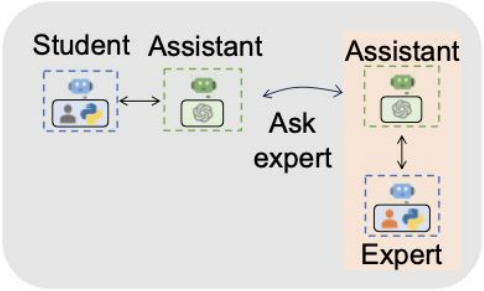

AutoGen Agents
ConversableAgent
Agent Customization:
human_input_mode = "NEVER"
code_execution_config = False
DEFAULT_SYSTEM_MESSAGE = "You are a helpful AI assistant… In the following cases, suggest python code…"
Unified Conversation Interfaces:
send
receive
generate_reply
human_input_mode = "ALWAYS"
human_input_mode = "NEVER"
group_chat = [ ]
AssistantAgent
UserProxyAgent
GroupChatManager
Developer Code
1.2 Register a Custom Reply Func:
# This func will be invoked in generate_reply
A.register_reply(B, reply_func_A2B)
def reply_func_A2B(msg):
ouput = input_from_human()
if not ouput:
if msg includes code:
output = execute(msg)
return output
1.1 Define Agents:
User Proxy A
Assistant B
Note: when no reply func is registered, a list of default reply functions will be used.
2 Initiate Conversations:
A.initiate_chat("Plot a chart of META and TESLA stock price change YTD.", B)
Program Execution
Conversation-Driven Control Flow
The Resulting Automated Agent Chat:
Plot a chart of META and TESLA stock price change YTD.
receive
generate_reply
send
Execute the following code...
receive
generate_reply
Error: package yfinance is not installed
send
generate_reply
Sorry! Please first pip install yfinance and then execute
Conversation-Centric Computation

Applications of AutoGen

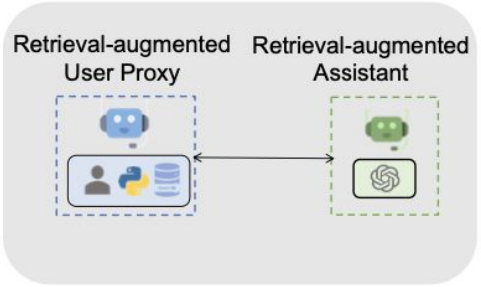


A1. Math Problem Solving

Applications of AutoGen

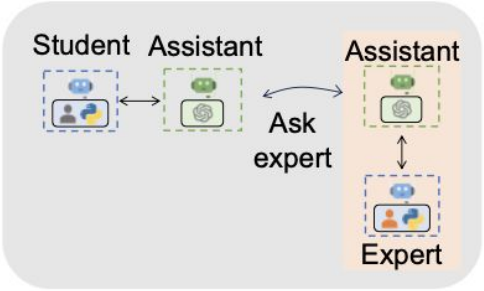


A1. Math Problem Solving

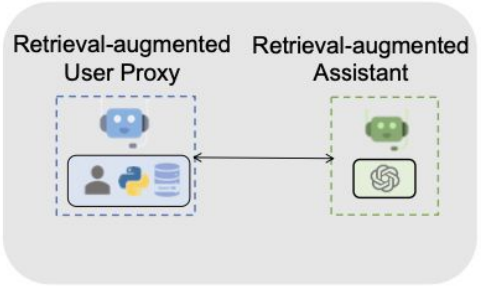


A2. Retrieval-augmented Chat

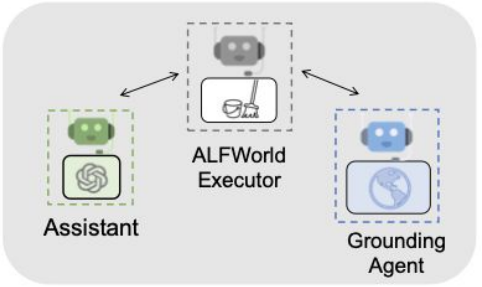
Applications of AutoGen



A1. Math Problem Solving

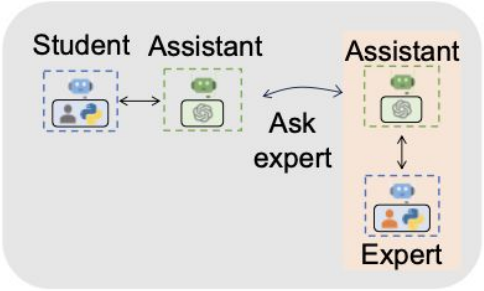


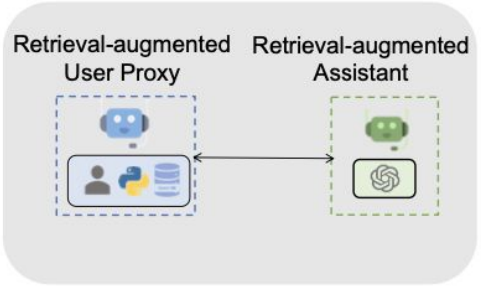
A2. Retrieval-augmented Chat



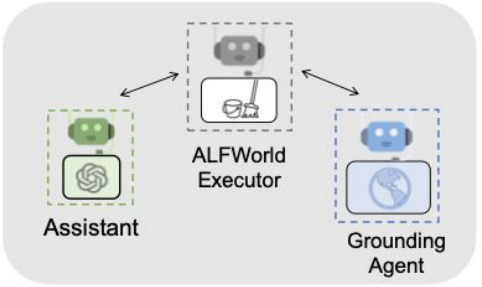
A3. ALF Chat

Applications of AutoGen

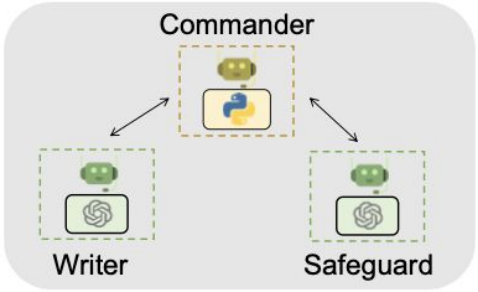




A2. Retrieval-augmented Chat

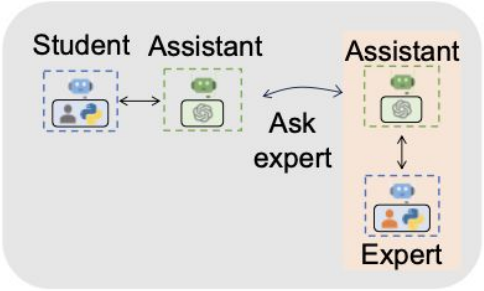


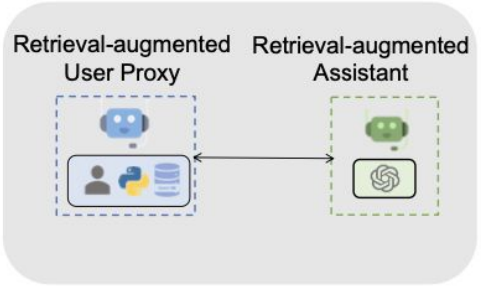
A3. ALF Chat

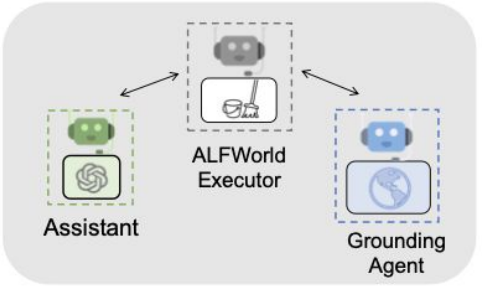


A4. Multi-agent Coding

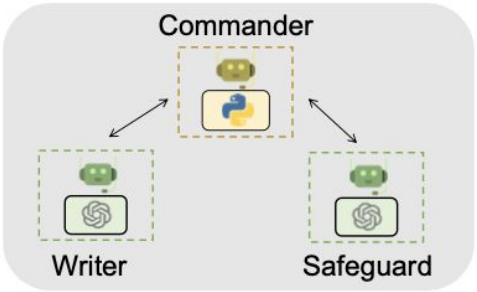
Applications of AutoGen



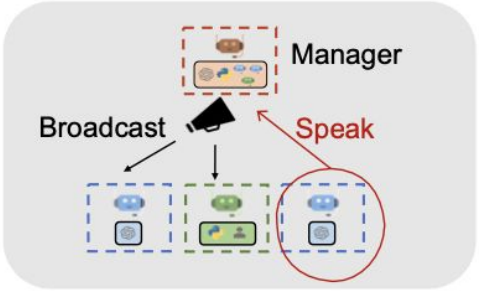




A3. ALF Chat

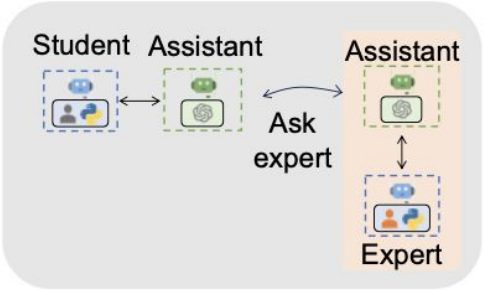


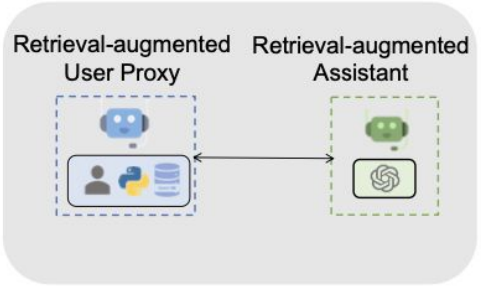
A4. Multi-agent Coding

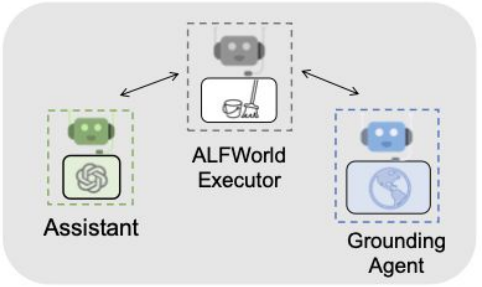


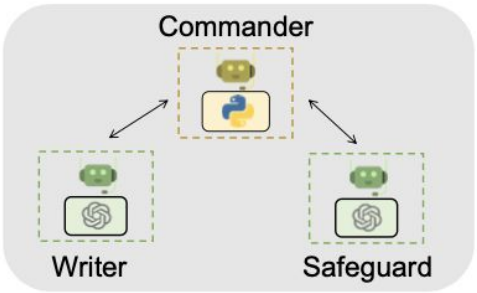
A5. Dynamic Group Chat

Applications of AutoGen

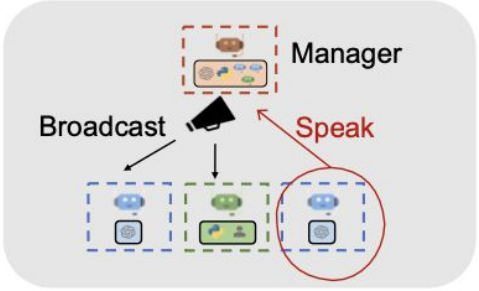




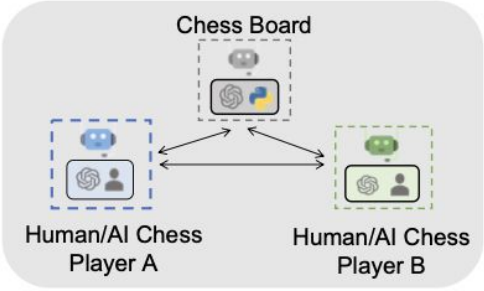




A4. Multi-agent Coding

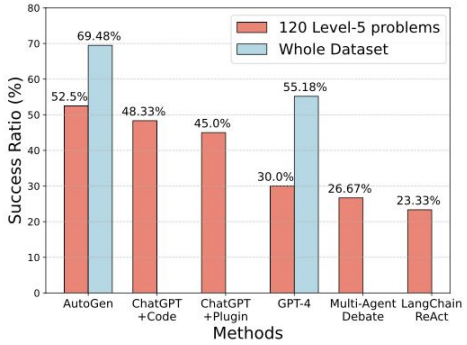


A5. Dynamic Group Chat

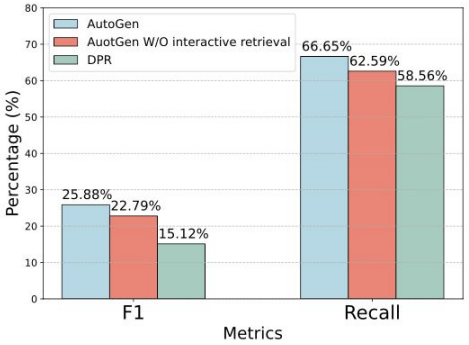


A6. Conversational Chess

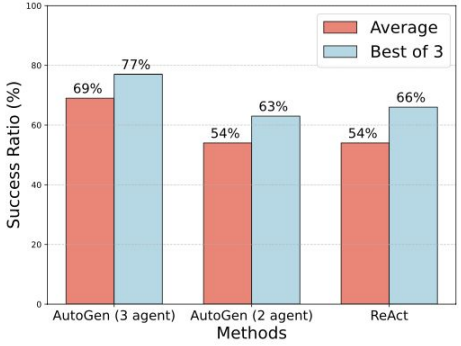
Results



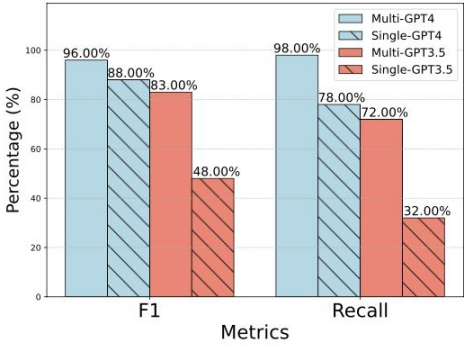
(a) A1: Performance on MATH (w/ GPT-4).



(b) A2: Q&A tasks (w/ GPT-3.5).



(c) A3: Performance on ALFWorld.



(d) A4: Performance on OptiGuide.

Summary

AutoGen is a general framework for building LLM applications using **multi-agent conversations**.

It provides:

- Modular, customizable agents;
- A programming interface that blends natural language and Python;
- Native support for tools, humans, and complex interaction patterns.

Each agent is “conversable”: it talks, thinks, and acts through messages.
The conversation is the program.

Multi-Agent Collaboration & Cooperation

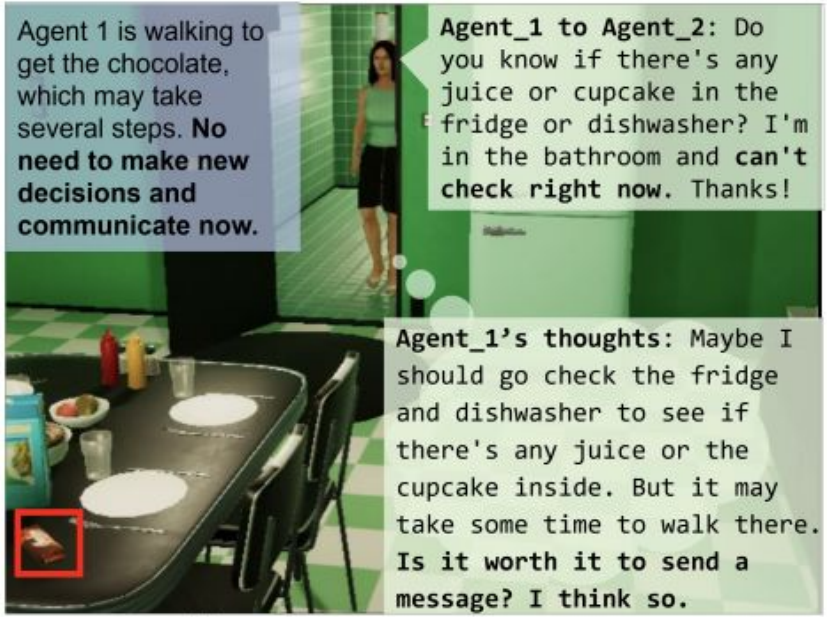
What happens when we put multiple LLMs in a room and ask them to collaborate?

Multi-Agent Cooperation

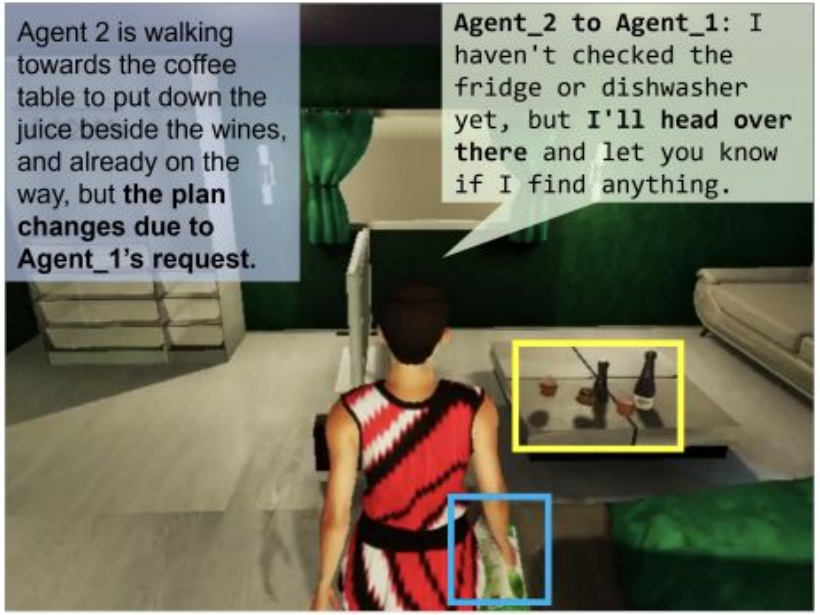
Embodied LLM Agents Learn to Cooperate in Organized Teams

Xudong Guo[1] Kaixuan Huang[2] Jiale Liu[3] Wenhui Fan[1] Natalia Vélez[2]
Qingyun Wu[3] Huazheng Wang[4] Thomas L. Griffiths[2] Mengdi Wang[2]
[1]Tsinghua University [2]Princeton University
[3]Penn State University [4]Oregon State University

Without any organization, agents don't cooperate well



Unnecessary messages



Plan disruption

https://arxiv.org/abs/2403.12482

Challenges

LLMs are great at individual reasoning and planning, current multi-agent setups don’t scale well:

- They often assume two-agent scenarios,
- They use hardcoded interaction patterns,
- They don’t support organizational flexibility
- Blindly follow instructions and over-communicate

Research Questions

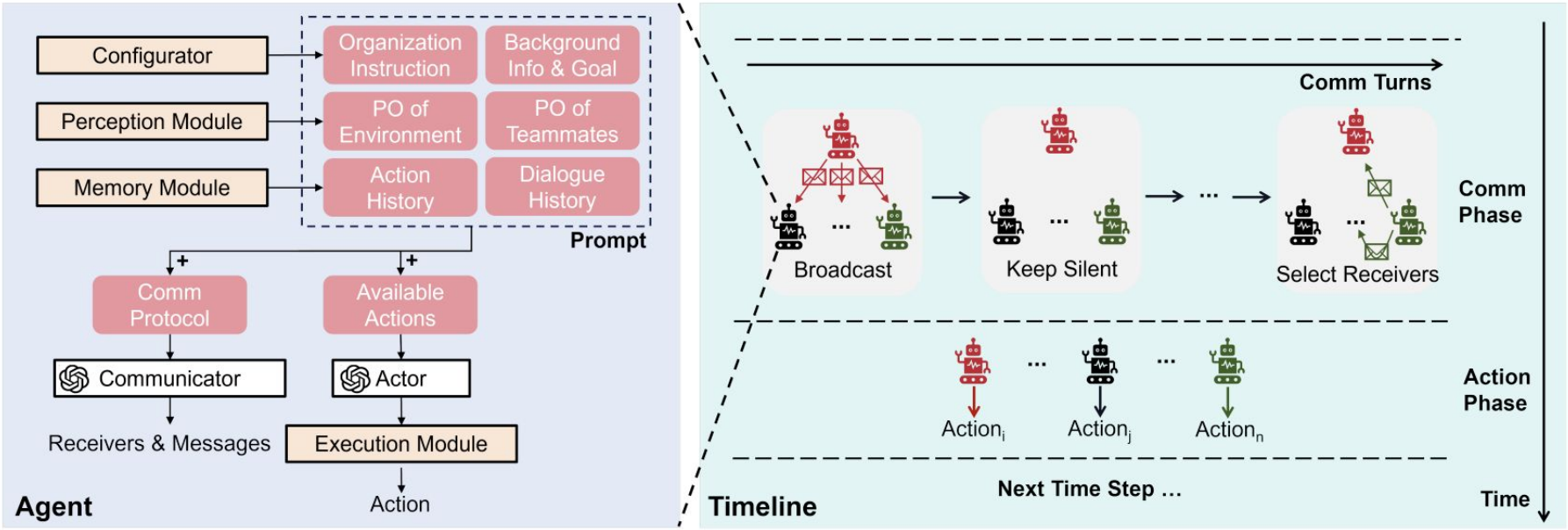
What role do organizational prompts play in LLM agent collaboration?

Research Questions

What role do organizational prompts play in LLM agent collaboration?

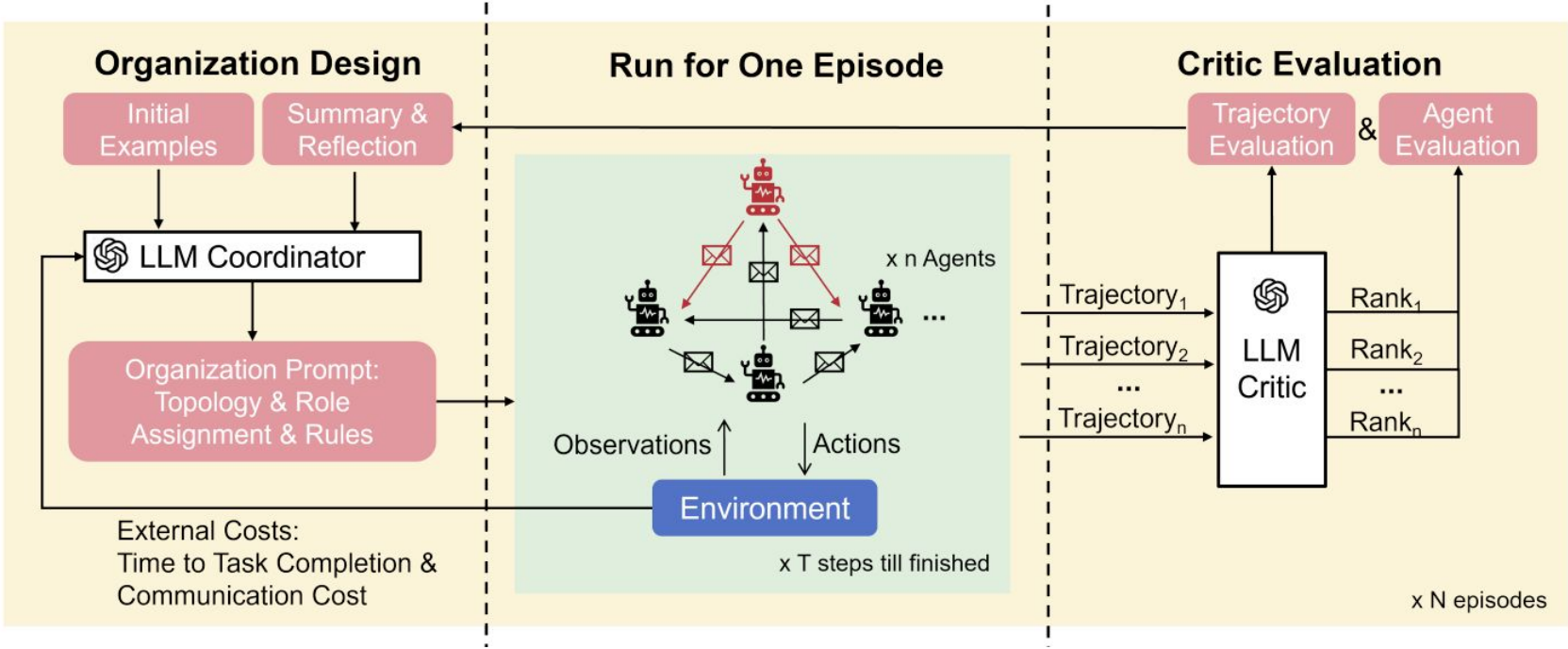
Can we let LLMs reflect on their performance and optimize those prompts over time?

Multi-LLM-RL Agents: Architecture



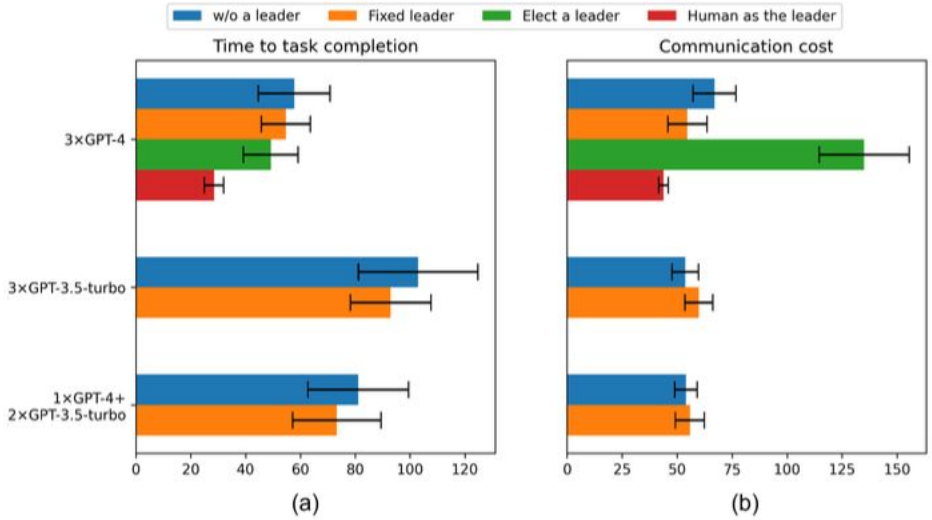
https://arxiv.org/abs/2403.12482

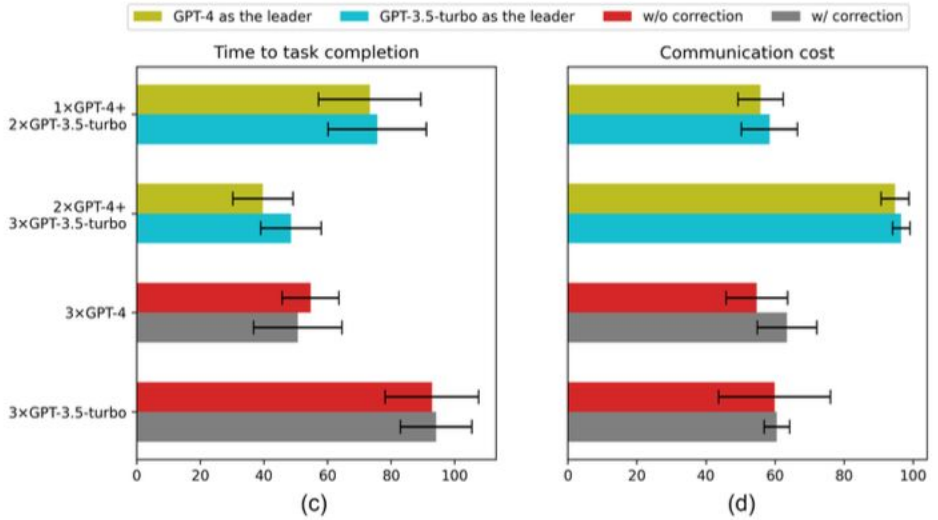
Self improving via LLM critic



https://arxiv.org/abs/2403.12482

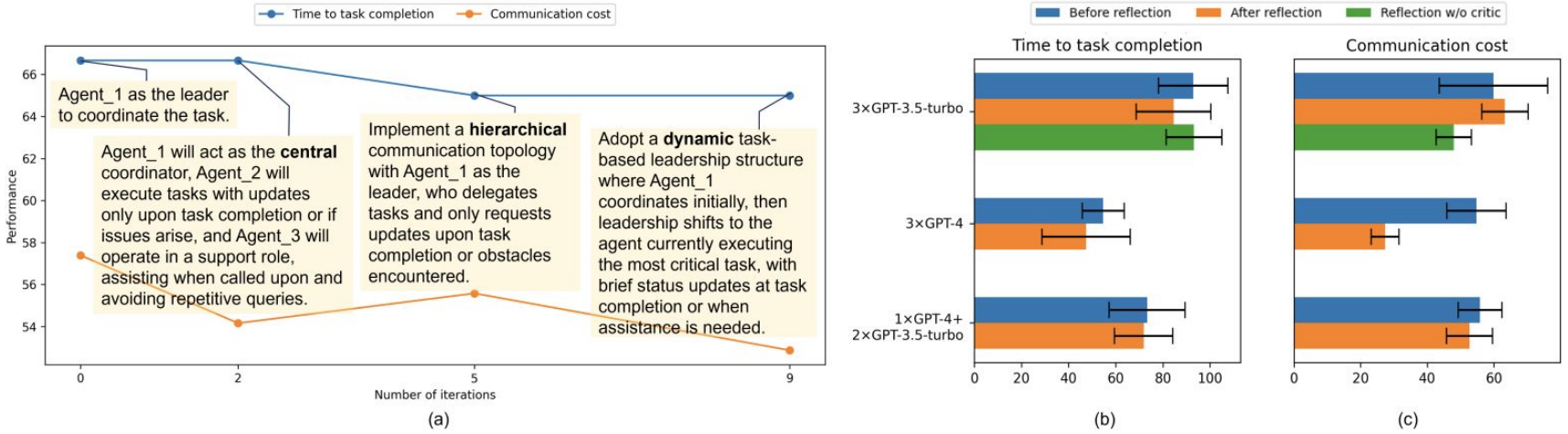
Efficiency





https://arxiv.org/abs/2403.12482

Emergence of Organizational Structures



https://arxiv.org/abs/2403.12482

Summary

- Structured inter-agent communication is essential for cooperative behavior.
- The integration of an LLM critic can greatly enhance coordination, leading to more robust and coherent team behavior.

Thank you!