Artificial Intelligence

Xiangmeng (Mona)WANG
(Research Assistant Professor)
Department of Computing
The Hong Kong Polytechnic University

Learning Outcomes of Lecture 02

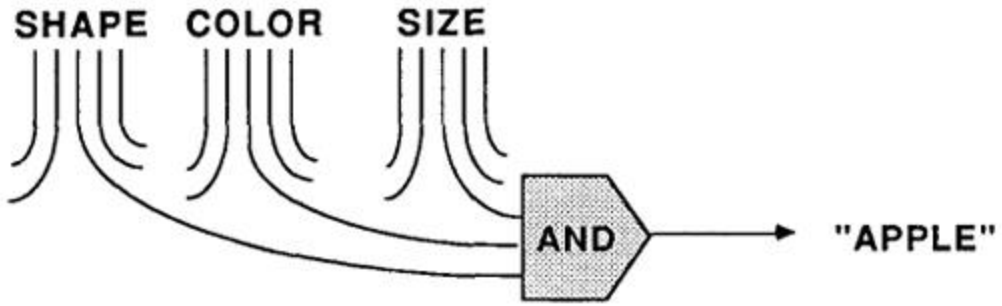
- **Symbolic Systems & Expert Systems**
  - Symbolic Systems: AI systems that represent knowledge as symbols and manipulate them with formal logic to perform reasoning
  - Transduction Problem: Converting raw sensory data (from the real world) into symbolic representations that the agent can use
  - Symbolic Reasoning: Manipulating symbols that represent concepts, objects, and rules to derive new knowledge
  - Expert System: Software that uses a knowledge base of human expertise for problem solving

Learning Outcomes of Lecture 03

- **Rational agent & Search agent**
  - Rational agent: a new definition of agent-anything that perceives its environment through sensors and acts upon that environment with actuators
  - PEAS: Performance measure; Environment ; Actuators; Sensors
  - Uninformed search - Breadth-first Search and Depth-first Search
  - Informed search - A* search

Timeline of Agent Research in AI

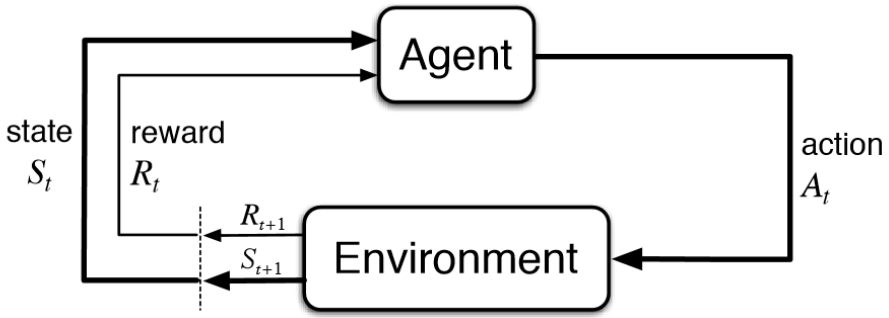
- The evolution of AI agents has undergone several stages
  - Symbolic agent stage: symbolic agent approaches employ logical rules and symbolic representations to encapsulate knowledge and facilitate reasoning processes.
  - A classic example of this approach is knowledge-based expert systems.



Minsky M. Society of mind[M]. Simon and Schuster, 1988.

Timeline of Agent Research in AI

- The evolution of AI agents has undergone several stages
  - Reinforcement learning-based agent stage: reinforcement learning-based agents utilize reinforcement learning methods to train agents to learn through interactions with their environments, enabling them to achieve maximum cumulative rewards in specific tasks.



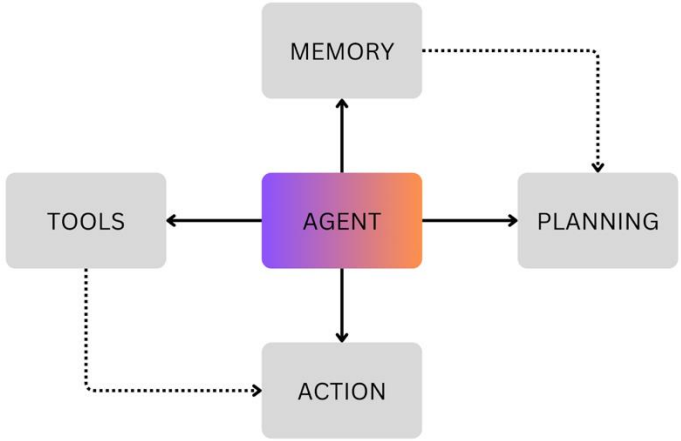
- The agent-environment interaction in reinforcement learning
  - An agent (decision maker) takes the action A_t in the given state s_t and receives the reward R_t.
  - After the reward is received, the agent update their parameters such that the total maximizing discounted expected reward.

Sutton R S, Barto A G. Reinforcement learning: An introduction[M]. MIT press, 2018.

Timeline of Agent Research in AI

- The evolution of AI agents has undergone several stages
  - Large language model-based agent stage: employ large language model as the primary component of brain or controller of agents.
    - LLMs can exhibit reasoning and planning abilities comparable to symbolic agents
    - LLMs can acquire interactive capabilities with the environment, by learning with feedback
    - LLMs undergo pre-training and demonstrate the capacity for few- and zero-shot generalization

  - Several key components in LLM
    - Memory: The primary function of the memory system is to preserve and regulate knowledge, experiential data, and historical information.
    - Planning: The planning capability defines an LLM-based agent's ability to devise action sequences based on set objectives and existing environment constraints, ensuring goal fulfillment.
    - Tool use: encompassing APIs, calculators, code interpreters, or actions within a physical environment through text-based directives.



Agenda

Session 1. Reinforcement Learning

- What is Reinforcement Learning?
- Markov Decision Processes
- Q-Learning
- Policy Gradients

Session 2. Agentic AI

- Agentic AI definition
- Key Requirements of Agentic AI
- LLMs for Agentic AI: Evolution
- Key Components of AI Agents

Agenda

Session 1. Reinforcement Learning

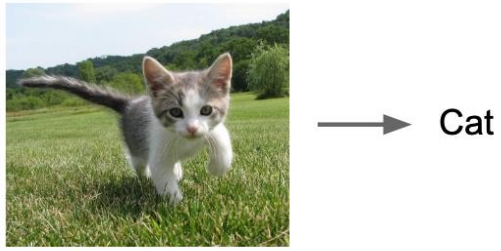
- What is Reinforcement Learning?
- Markov Decision Processes
- Q-Learning
- Policy Gradients

Session 2. Agentic AI

- Agentic AI definition
- Key Requirements of Agentic AI
- LLMs for Agentic AI: Evolution
- Key Components of AI Agents

Reinforcement Learning: overview

- Past paradigm: *Supervised Learning*
- Data: *(x, y)*
  - *x* is data, *y* is label
- Goal: Learn a function to map *x -> y*
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



Classification

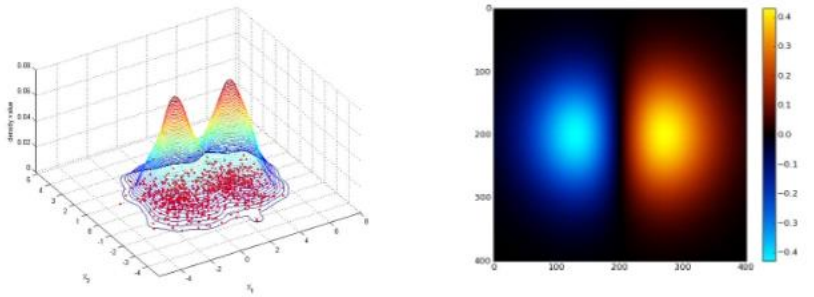
Reinforcement Learning: overview

- Past paradigm: *Unsupervised Learning*
- Data: x
  - Just data, no labels!
- Goal: Learn some underlying hidden structure of the data
- Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.



Figure copyright Ian Goodfellow, 2016. Reproduced with permission.

1-d density estimation



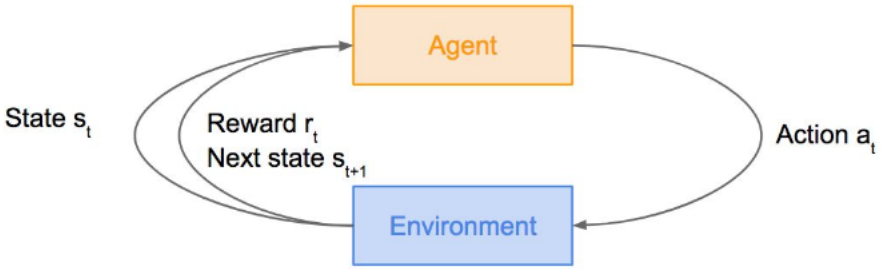
2-d density estimation

Reinforcement Learning: overview

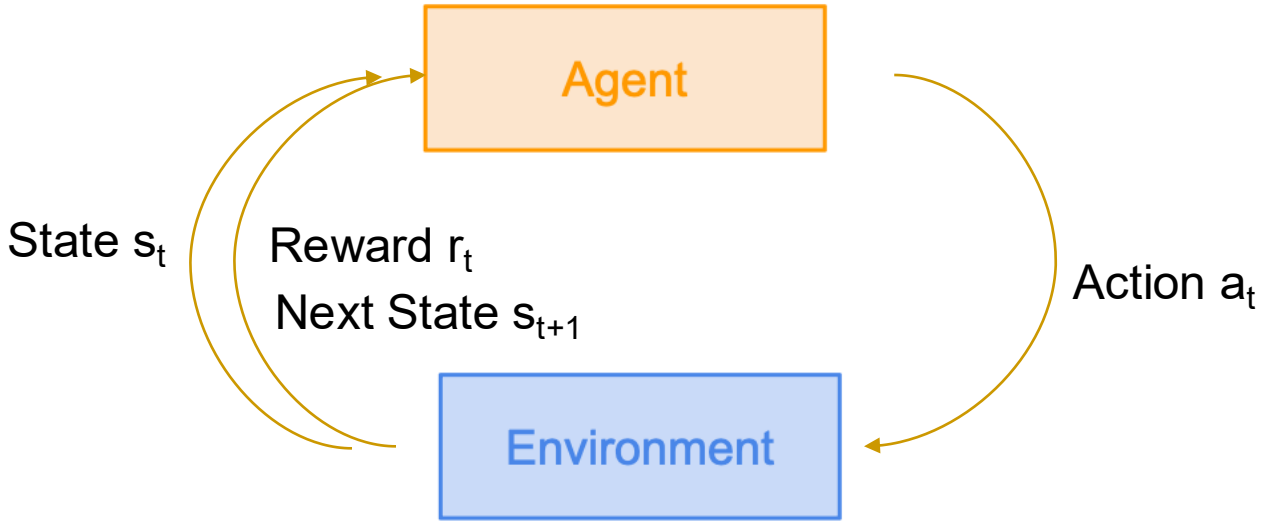
- Today: *Reinforcement Learning*

- Problems involving an **agent** interacting with an **environment**, which provides numeric **reward** signals

- Goal: Learn how to take actions in order to maximize reward

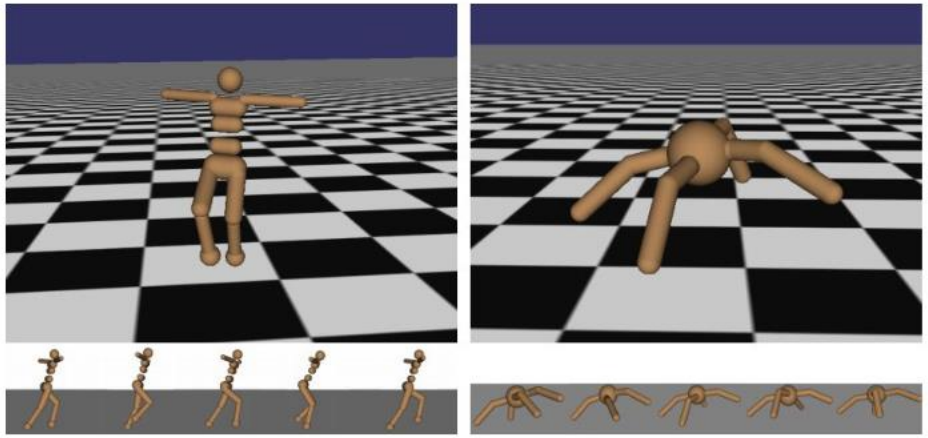


Reinforcement Learning



Reinforcement Learning

- Robot Locomotion
- Objective: Make the robot move forward

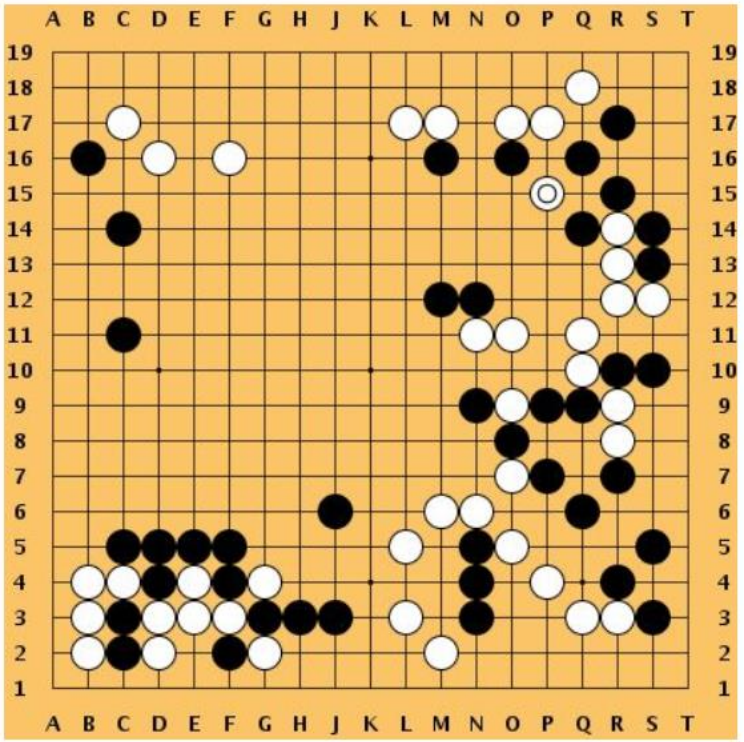


- State: Angle and position of the joints
- Action: Torques applied on joints
- Reward: 1 at each time step upright + forward movement

Reinforcement Learning

- Go
- Objective: Win the game!

- State: Position of all pieces
- Action: Where to put the next piece down
- Reward: 1 if win at the end of the game, 0 otherwise



Markov Decision Process

- How can we mathematically formalize the RL problem?
- Markov Decision Process: Mathematical formulation of the RL problem
  - Markov property (assumption): Current state completely characterises the state of the world
- Defined by: $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathbb{P}, \gamma)$

$\mathcal{S}$: set of possible states
$\mathcal{A}$: set of possible actions
$\mathcal{R}$: distribution of reward given (state, action) pair
$\mathbb{P}$: transition probability i.e. distribution over next state given (state, action) pair
γ : discount factor

Markov Decision Process

- At time step t=0, environment samples initial state $s_0 \sim p(s_0)$
- Then, for t=0 until done:
  - Agent selects action a_t
  - Environment samples reward $r_t \sim \mathbf{R}(. \mid s_t, a_t)$
  - Environment samples next state $s_{t+1} \sim \mathbf{P}(. \mid s_t, a_t)$
  - Agent receives reward rt and next state s_{t+1}
- A policy is a function from **S** to **A** that specifies what action to take in each state
- Objective: find policy π^* that maximizes cumulative discounted reward: $\sum_{t>0} \gamma^t r_t$

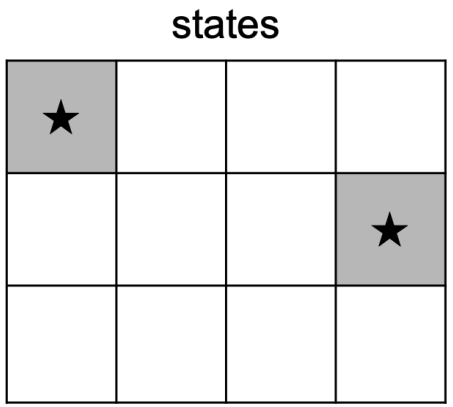
Markov Decision Process

- A simple MDP: Grid World

actions = {

1. right
2. left
3. up
4. down

}

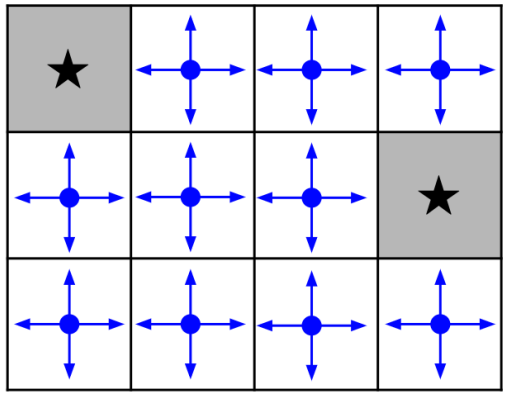


Set a negative "reward" for each transition (e.g. r = -1)

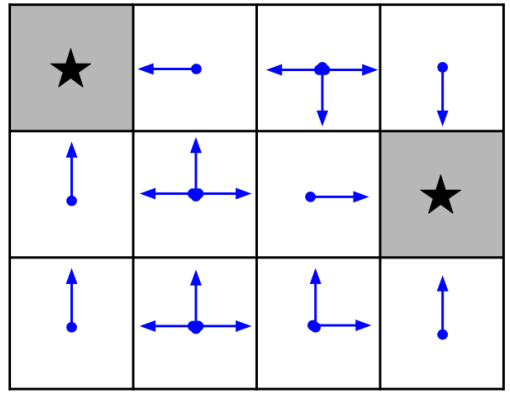
Objective: reach one of terminal states (greyed out) in least number of actions

Markov Decision Process

- Solution:



Random Policy



Optimal Policy

The optimal policy π^*

- We want to find optimal policy π^* that maximizes **the sum of rewards**.

- But… How do we handle the randomness (initial state, transition probability…)?
  - Remember the "Rationality is not omniscience" we learnt in last lecture?

The optimal policy π^*

- The solution: Maximize **the expected sum of rewards**!

- Formally: $$\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t \geq 0} \gamma^t r_t \mid \pi\right] \text{ with } s_0 \sim p(s_0), a_t \sim \pi(\cdot \mid s_t), s_{t+1} \sim p(\cdot \mid s_t, a_t)$$

Value function and Q-value function

- Following a policy produces sample trajectories (or paths)

$$s_0, a_0, r_0, s_1, a_1, r_1, \ldots$$

Measurement: How good is a state?

The value function at state s, is the **expected cumulative reward** from *following the policy from state s*:

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t \geq 0} \gamma^t r_t | s_0 = s, \pi\right]$$

Measurement: How good is a state-action pair?

The Q-value function at state s and action a, is the **expected cumulative reward** from *taking action a in state s* (i.e., state-action pair) and then *following the policy*:

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t \geq 0} \gamma^t r_t | s_0 = s, a_0 = a, \pi\right]$$

Bellman equation

- Having the measurement for state-action pair (i.e., Q- value function), the goal becomes: finding the optimal Q-value function Q*:
  - i.e., the maximum expected cumulative reward achievable from a given (state, action) pair

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | s_0 = s, a_0 = a, \pi\right]$$

- Bellman equation:
  - Breaks down a long-term reward problem into smaller, solvable subproblems.
- Q* satisfies the following Bellman equation:

$$Q^*(s,a) = \mathbb{E}_{s'\sim\mathcal{E}}\left[r + \gamma \max_{a'} Q^*(s',a') | s,a\right]$$

Intuition: if the optimal state-action values for the **next time-step Q*(s',a')** are known, then the optimal strategy is to take the action that maximizes the expected value of $r + \gamma Q^*(s',a')$

Solving for the optimal policy

- Bellman equation indicate: The optimal policy π * corresponds to taking the best action in any state as specified by Q*

- Solution:
  - Value iteration algorithm: Use Bellman equation as an iterative update

$$Q_{i+1}(s,a) = \mathbb{E}\left[r + \gamma \max_{a'} Q_i(s',a') | s,a\right]$$

Qi will converge to Q* as i -> infinity

What's the problem with this?

Not scalable! Must compute Q(s, a) for every state-action pair.

If state is e.g. current game state pixels, computationally infeasible to compute for entire state space!

Q-learning

- Scalability Solution: use a function approximator to estimate Q(s, a). e.g. a neural network!

- Q-learning: Use a function approximator to estimate the action-value function

function parameters (weights)

$$Q(s, a; \theta) \approx Q^*(s, a)$$

- If the function approximator is a deep neural network => deep q-learning!

Q Network Training

- Remember: want to find a Q-function that satisfies the Bellman Equation:

$$Q^*(s,a) = \mathbb{E}_{s' \sim \mathcal{E}}\left[r + \gamma \max_{a'} Q^*(s',a') | s, a\right]$$

- Forward pass of NN:

NN with function parameters

Loss function: $L_i(\theta_i) = \mathbb{E}_{s,a \sim \rho(\cdot)}\left[(y_i - Q(s,a;\theta_i))^2\right]$

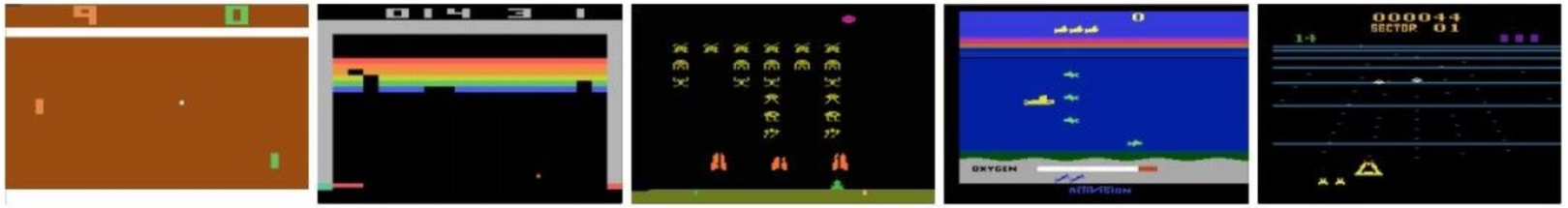
where $y_i = \mathbb{E}_{s' \sim \mathcal{E}}\left[r + \gamma \max_{a'} Q(s',a';\theta_{i-1}) | s, a\right]$

Iteratively try to make the Q-value close to the target value (y_i) it should have, if Q-function corresponds to optimal Q* (and optimal policy *)
MSE!

- Backward Pass
  - Gradient update (with respect to Q-function parameters θ):

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}_{s,a \sim \rho(\cdot); s' \sim \mathcal{E}}\left[r + \gamma \max_{a'} Q(s',a';\theta_{i-1}) - Q(s,a;\theta_i)) \nabla_{\theta_i} Q(s,a;\theta_i)\right]$$

Case Study: Playing Atari Games



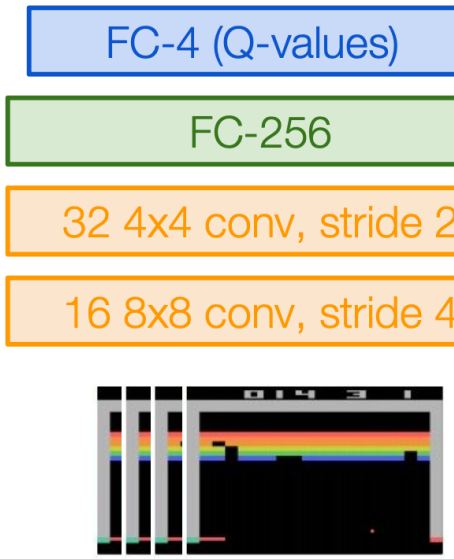
- Objective: Complete the game with the highest score

- State: Raw pixel inputs of the game state
- Action: Game controls e.g. Left, Right, Up, Down
- Reward: Score increase/decrease at each time step

Q-network Architecture

$Q(s, a; \theta)$:
neural network
with weights θ

A single feedforward pass to compute Q-values for all actions from the current state => efficient!



Last FC layer has 4-d output (if 4 actions), corresponding to $Q(s_t, a_1)$, $Q(s_t, a_2)$, $Q(s_t, a_3)$, $Q(s_t, a_4)$

Familiar conv layers, FC layer

Input: state s_t

Current state s_t: 84x84x4 stack of last 4 frames
(after RGB->grayscale conversion, downsampling, and cropping)

[Mnih et al. NIPS Workshop 2013; Nature 2015]

Other Resources

- Policy Gradients:
  - What is a problem with Q-learning?
    - The Q-function can be very complicated!
  - Example: a robot grasping an object has a very high-dimensional state => hard to learn exact value of every (state, action) pair
- But the policy can be much simpler:
  - Can we learn a policy directly, e.g. finding the best policy from a collection of policies?
    - a logging policy, a target policy
- Policy-based methods:
  - REINFORCE algorithm
- Off-policy and On-policy

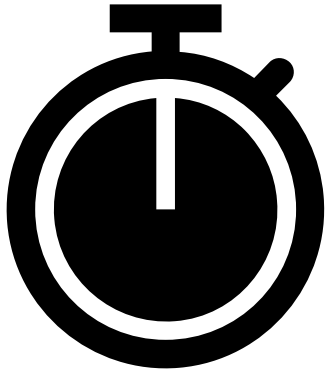
Other Resources

- Actor-Critic Algorithm
  - Problem: we don't know Q and V. Can we learn them?
  - Yes, using Q-learning! We can combine Policy Gradients and Q-learning
  - By training both an actor (the policy) and a critic (the Q-function)
    - The actor decides which action to take, and the critic tells the actor how good its action was and how it should adjust

- More policy gradients: AlphaGo
  - Overview:
    - Mix of supervised learning and reinforcement learning
    - Mix of old methods (Monte Carlo Tree Search) and recent ones (deep RL)

Quiz time

- You are allowed 10 min to accomplish the quiz
- After you finish, handle it to TA



Agenda

Session 1. Reinforcement Learning

- What is Reinforcement Learning?
- Markov Decision Processes
- Q-Learning
- Policy Gradients

Session 2. Agentic AI

- Agentic AI definition
- Key Requirements of Agentic AI
- LLMs for Agentic AI: Evolution
- Key Components of AI Agents

Agentic AI: overview

- AI agents vs. AI assistants

- Imagine you are a movie star or star footballer. You probably have an agent and an assistant.

Reactive!: You give Task

Your assistant does tasks for you, based on your requests:

- Make dinner reservations
- Pick up the dry cleaning
- Organize fan mail
- Help maintain your calendar

Proactive!: You give GOAL!

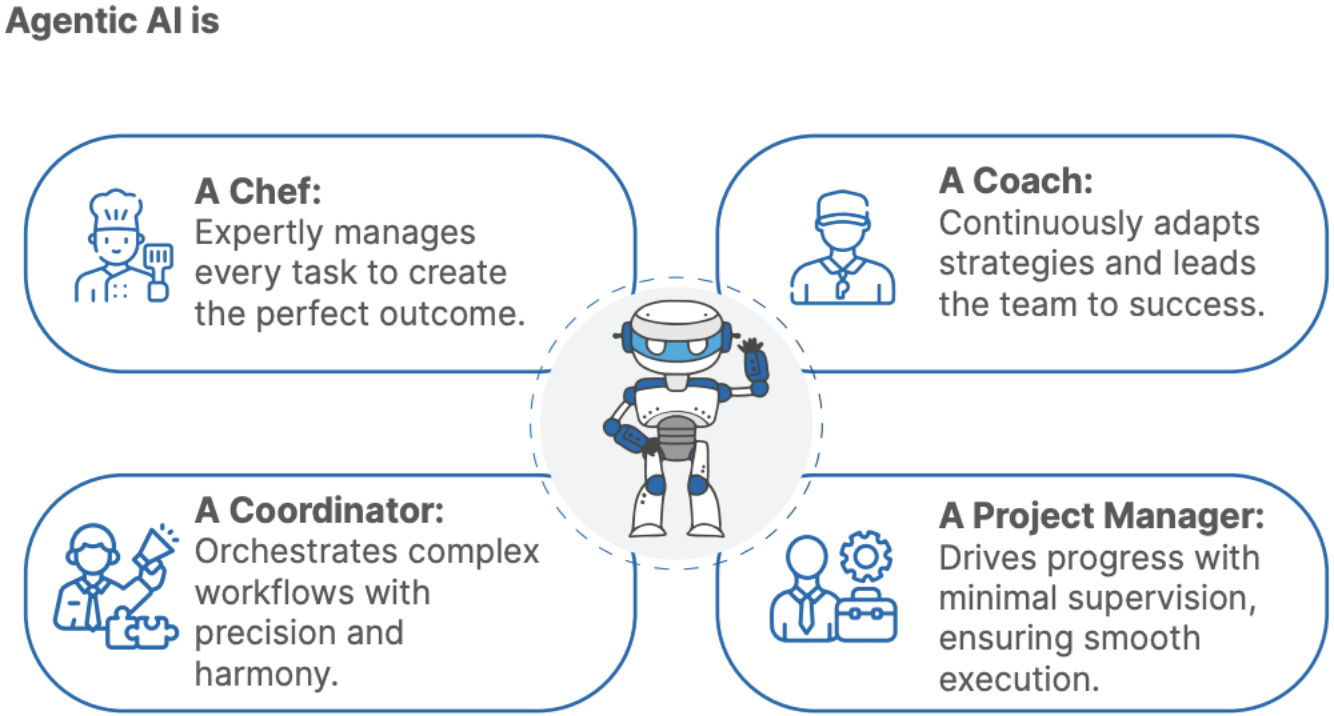
Your agents using their expertise day and night to maximize your opportunities and income:

- Plan your training to maximize your performance;
- Contact promoters

Agentic AI: overview

- What is Agentic AI?

- At its core, Agentic AI is about:
  - Understanding context beyond literal instructions
  - Breaking down complex goals
  - Making independent, autonomous decisions
  - Learning and adapting dynamically
  - Taking initiative without constant human supervision

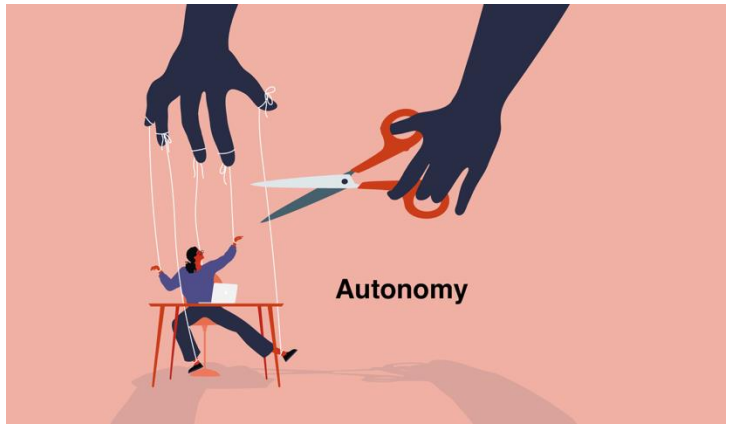


Agentic AI: overview

- Formal definition: An Agentic AI is a system that autonomously performs tasks by designing workflows with available tools.
- AI agents can encompass a wide range of functions beyond natural language processing including decision-making, problem-solving, interacting with external environments and performing actions.
- It's important to differentiate between Agentic AI and AI agents. Essentially, agentic AI is the framework; AI agents are the building blocks within the framework.

Key Requirements of Agentic AI

- The term “agentic” refers to these models’ agency, or, their capacity to act independently and purposefully.



Autonomy

Operate independently, making decisions based on their programming, learning, and environmental inputs.



Goal-oriented behaviour

designed to pursue specific objectives, optimising their actions to achieve the desired outcomes

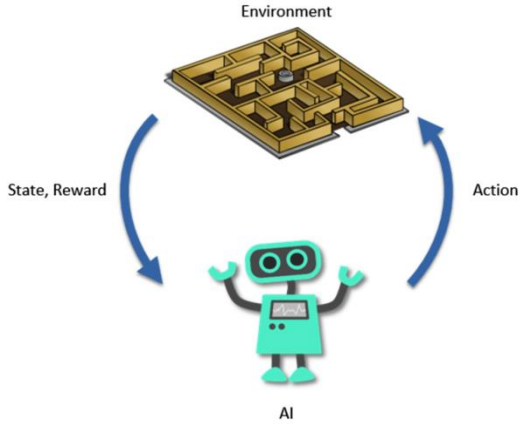


Environment interaction

interacts with its surroundings, perceiving changes and adapting its strategies accordingly.

Key Requirements of Agentic AI

- The term “agentic” refers to these models’ agency, or, their capacity to act independently and purposefully.



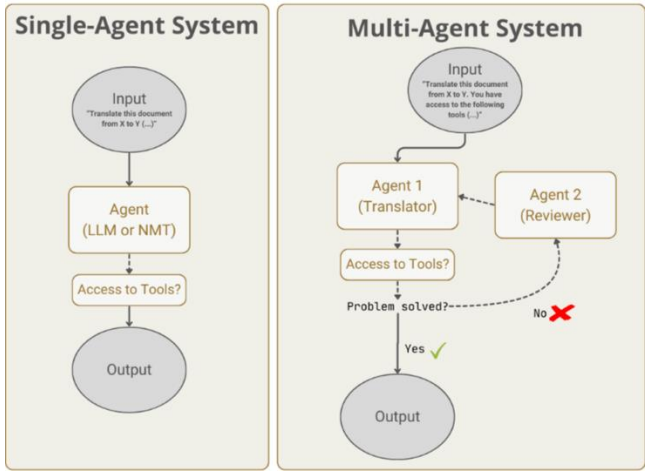
Learning capability

Many agentic AI systems employ machine learning or reinforcement learning techniques to improve their performance over time.



Workflow optimisation

Enhance workflows and business processes by integrating LU with reasoning, planning, and decision-making.

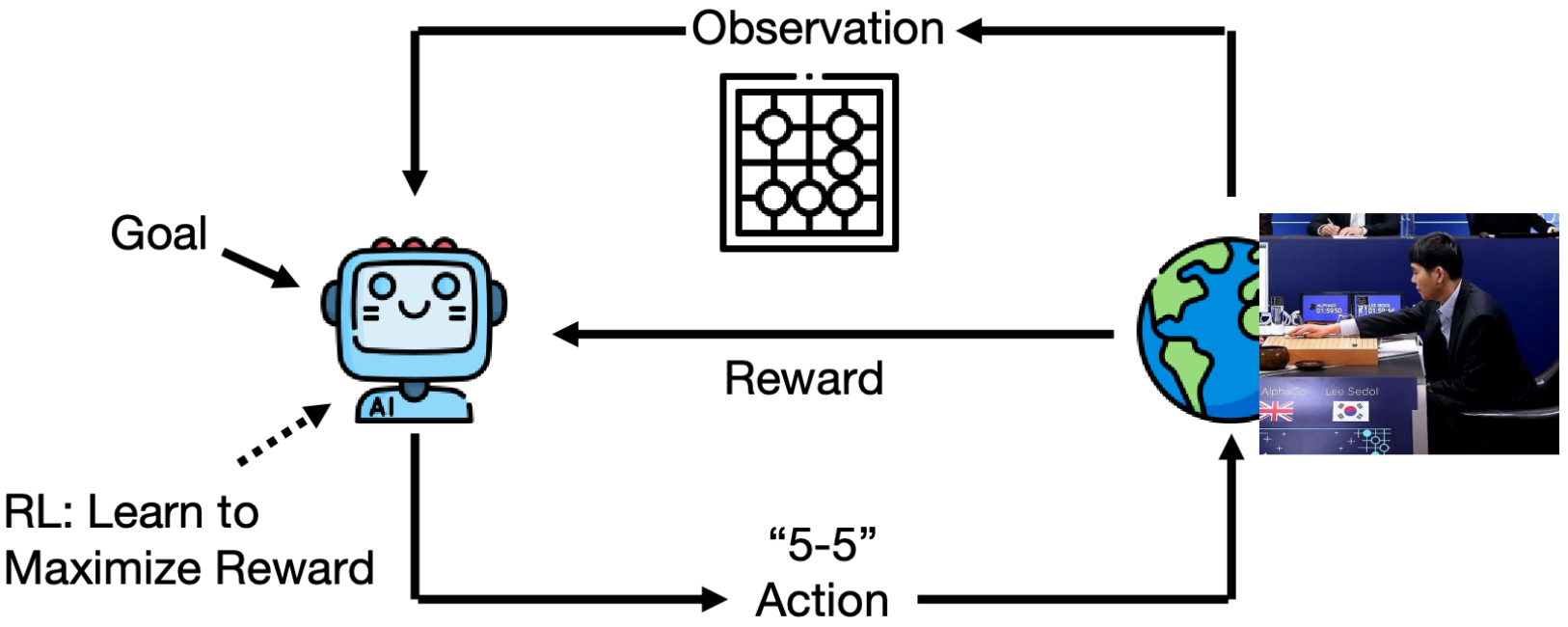


Multi-agent and system Conversation

Communication between different agents to construct complex workflows.

How to create AI Agent? RL?

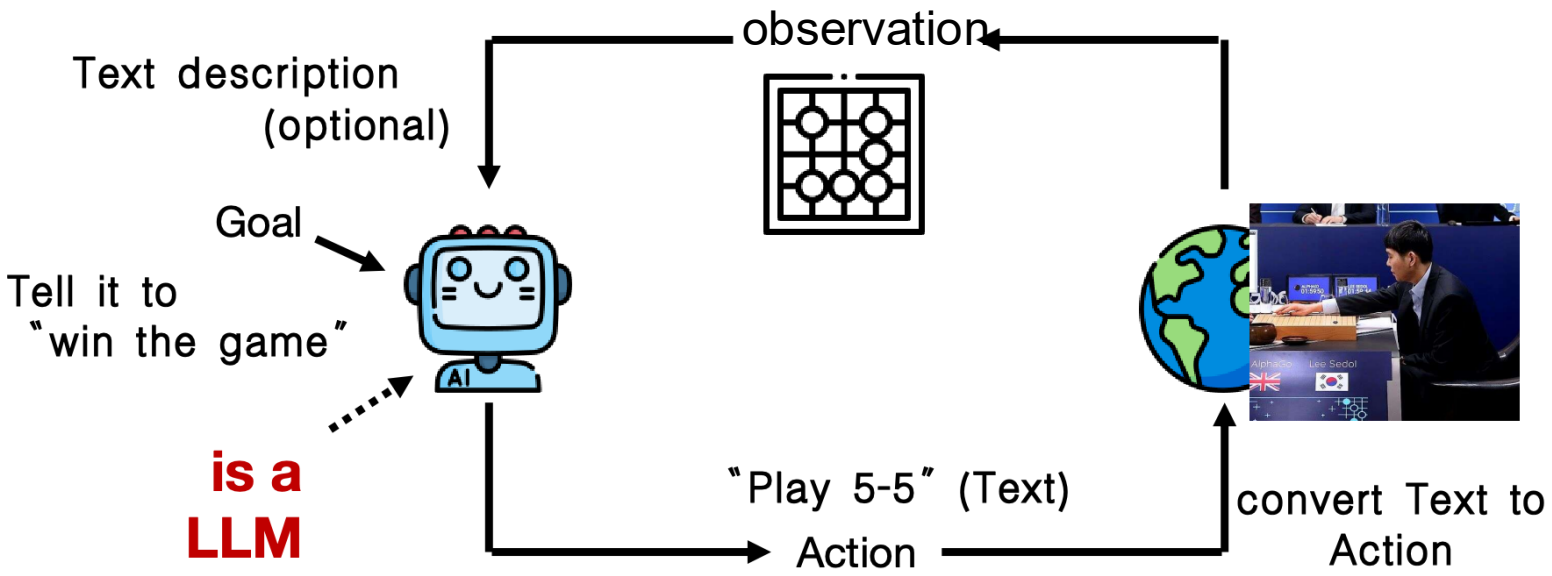
Revisiting AlphaGO:



Limitation:
Requires training an RL model for each task.

Create AI Agent: Embrace LLM!

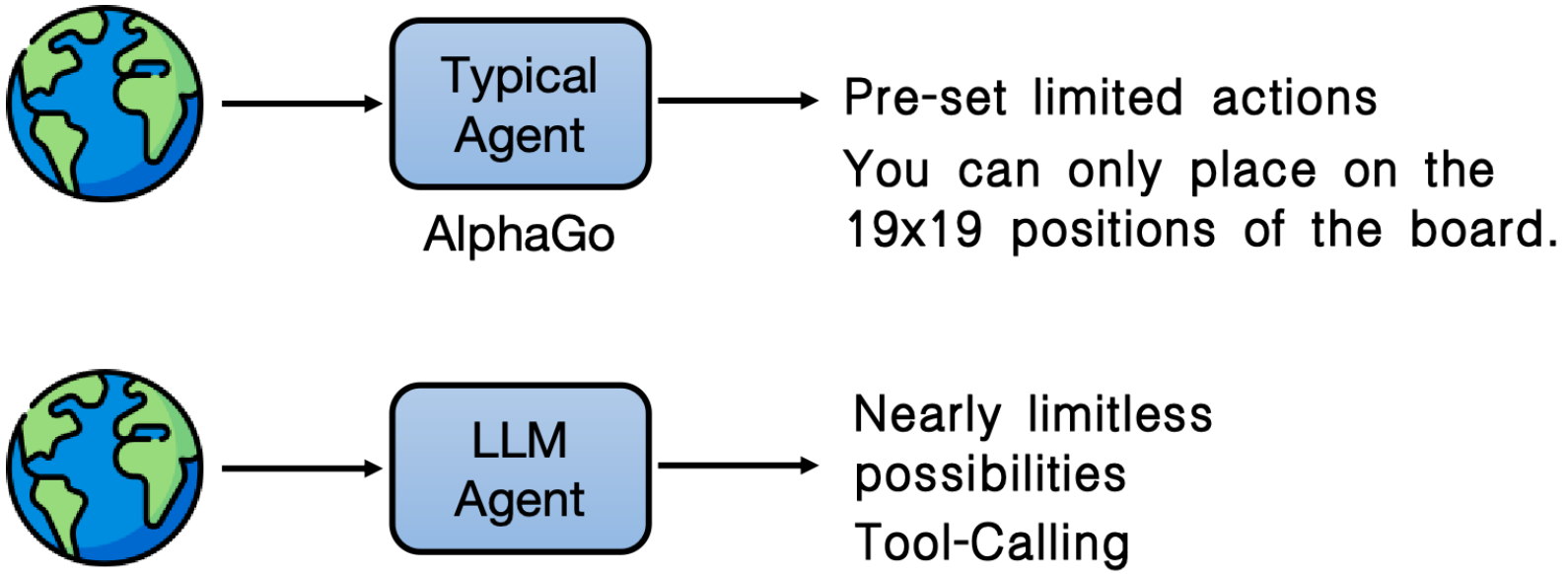
LLM as a Agent!



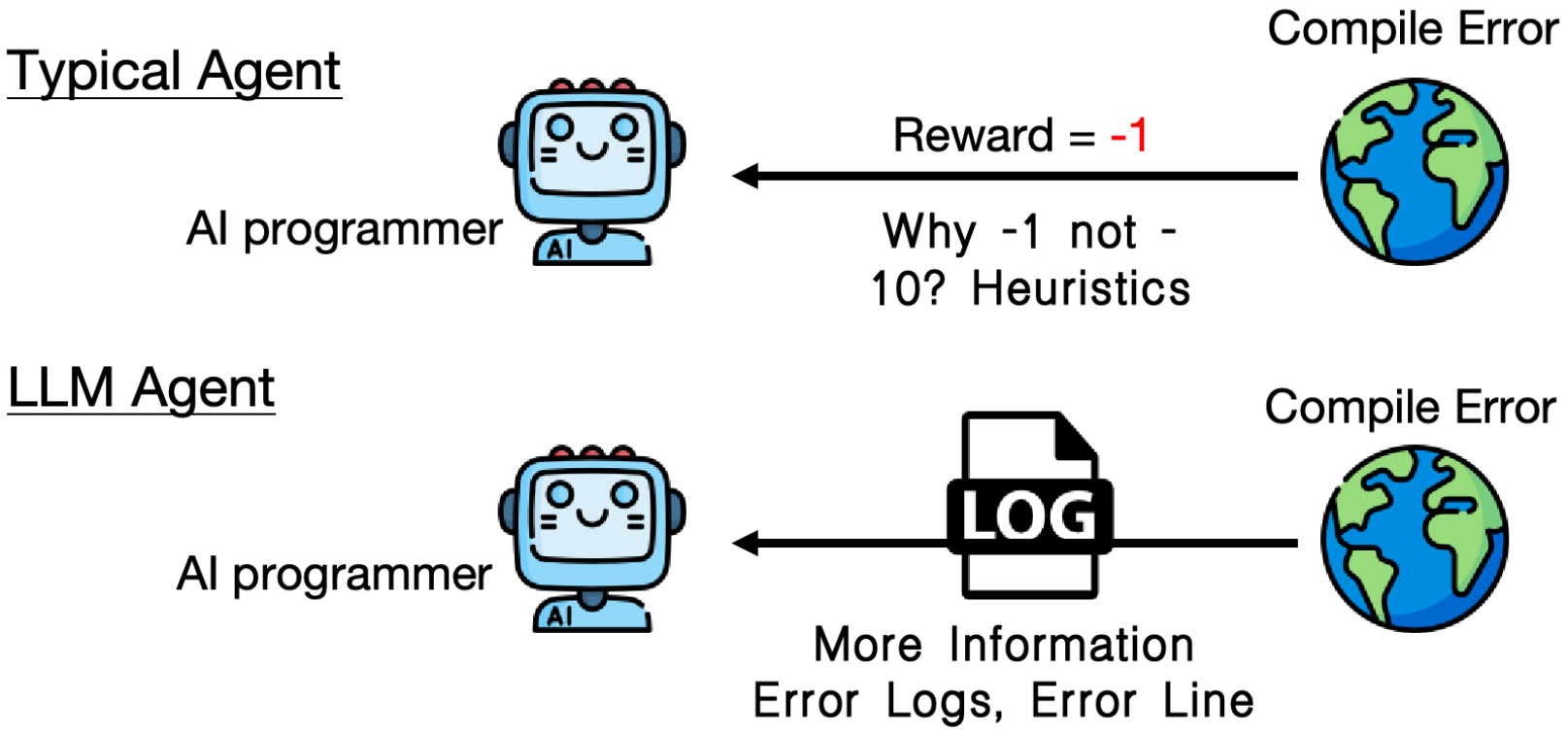
Using LLM to directly fulfill humanity's desire for agents

- The current Agentic AI hot topic is largely powered by the development of LLMs

Advantages of Using LLMs as Agents



Advantages of Using LLMs as Agents



Evolution to multimodal GenAI agents

- In AI, the only constant is change—embrace a culture of perpetual innovation.
- Integration of Machine Learning (ML)
- (2000s)
  - Learning from data: The integration of ML allowed agents to learn from large datasets, improving their ability to make decisions and perform tasks. This was *a significant step forward from rule-based systems*, as agents could now adapt to new information and improve over time.
  - Natural Language Processing (NLP) enabled user interactions: Advances in NLP enabled agents to *understand and generate human language* more effectively, making interactions more natural and intuitive.

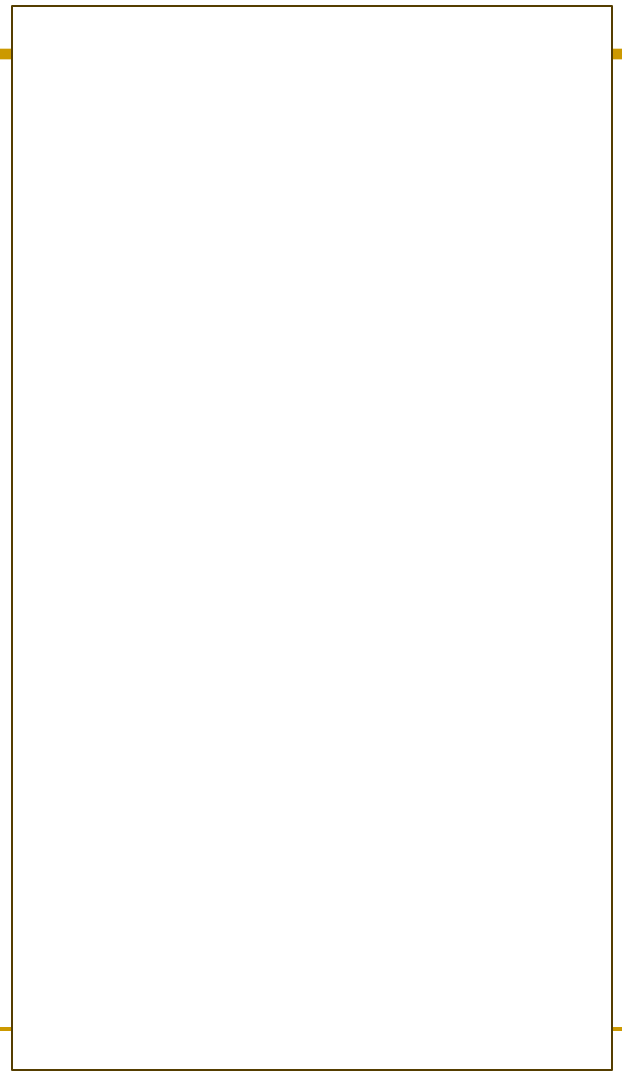
Evolution to multimodal GenAI agents

- In AI, the only constant is change—embrace a culture of perpetual innovation.
- Introduction of multimodality
- (2010s)
  - Multimodal agents emerged, capable of processing and integrating information from various sources. For instance, an agent could analyse a text description, recognise objects in an image, and understand spoken commands.
  - This multimodality made agents more versatile and capable of handling complex tasks.
  - Enhanced user interactions: Multimodal agents could interact with users in more dynamic ways, such as providing visual aids in response to text queries or understanding context from a combination of spoken and visual inputs.

Evolution to multimodal GenAI agents

- In AI, the only constant is change—embrace a culture of perpetual innovation.
- Advanced autonomy and real-time interactions
- (2020s-present)
  - Advancement of LLMs (GenAI): on top of Agentic AI
  - Advanced autonomy: Agents can operate independently, rationalise and set their own goals, develop path(s) to attain these goals, and make independent decisions without constant human intervention, leveraging data from multiple sources or synthetic datasets
  - User interactions within an ethical and responsible AI-controlled environment: With increased capabilities, there has also been a focus on ensuring that agentic systems operate ethically and responsibly, considering factors such as bias, transparency, and accountability.

A example of multimodal GenAI agents



What features can you see?

For patient:

- User profile generation
- Language supporting (English and Chinese)
- Suggestions for follow-up consultation
- Consultation calendar
- …

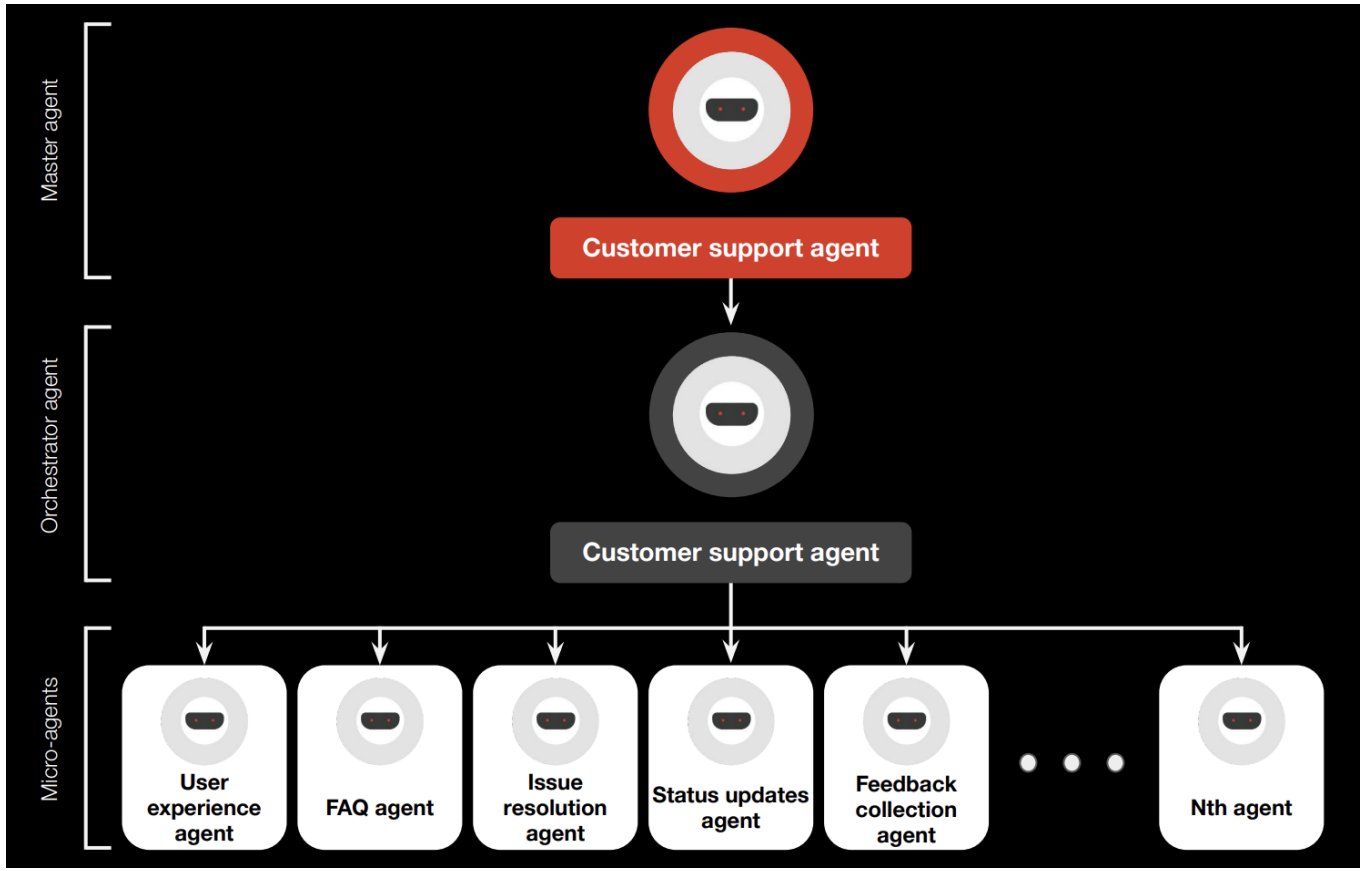
For doctor

- medicine treatment suggestion (mostly rule-based)
- Multimodal CT image Analyzing
- User query key information summarization
- …

How to conceptualise agentic AI solutions for future operations

Use case:

- **Education**
  - Personalized Learning: AI adapts learning materials to suit each student's progress and style, helping improve learning outcomes and engagement.
  - Automated Grading: AI agents handle grading of assignments and exams,
  - Others?



Challenges and Mitigation Strategies

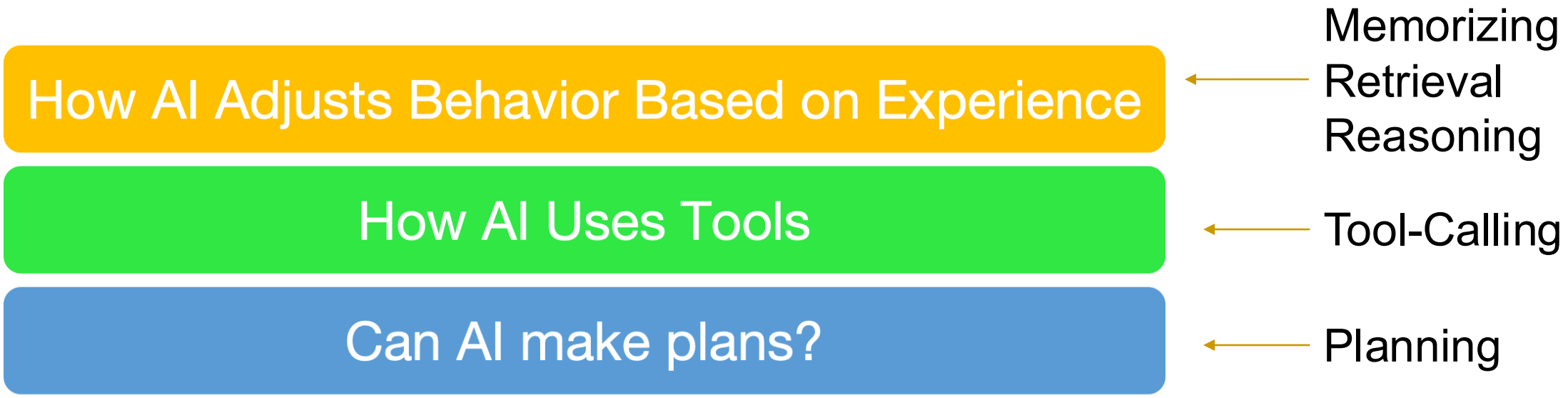
Challenge	Description	Mitigation Strategy
Complex System Design	Can become difficult to design due to inter-agent communication and dependencies.	Use modular agent frameworks to simplify agent development and deployment.
Interoperability	Integrating agents with existing ERP, WMS, and TMS systems can be difficult.	Adopt standardized APIs and data formats for seamless integration.
Data Security	Sensitive data shared between agents and external vendors can be vulnerable to breaches.	Implement end-to-end encryption, secure authentication protocols, for traceability.
Conflict Resolution	Agents may have competing priorities (e.g., logistics vs. inventory) leading to inefficiencies.	Introduce a decision-making agent with conflict resolution algorithms to mediate disputes.
Scalability	Scaling agents for large systems can strain computational resources and increase latency.	Use edge computing and cloud infrastructure for distributed processing and scalability.
Cost of Implementation	High initial investment in developing and deploying multi-agent systems.	Start with critical areas first; use incremental deployment and open-source frameworks to cut costs.

Agentic AI has a long way to go…

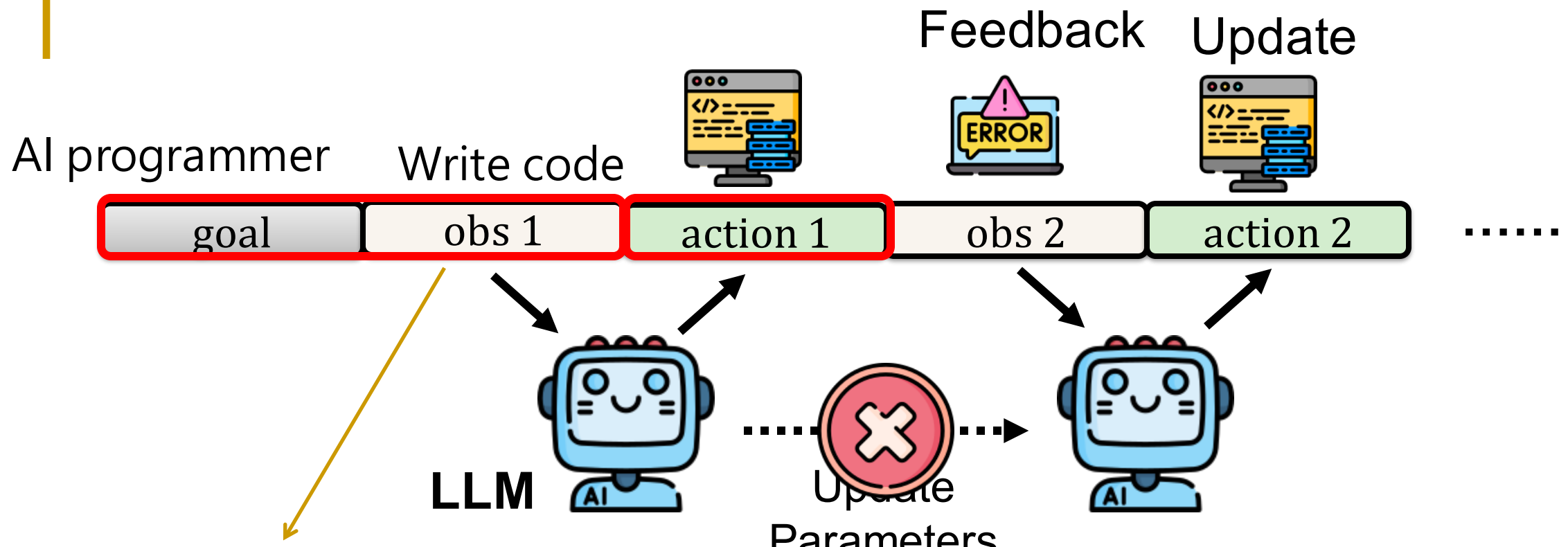
Current stage: Only a fantasy

Key Components of AI Agents

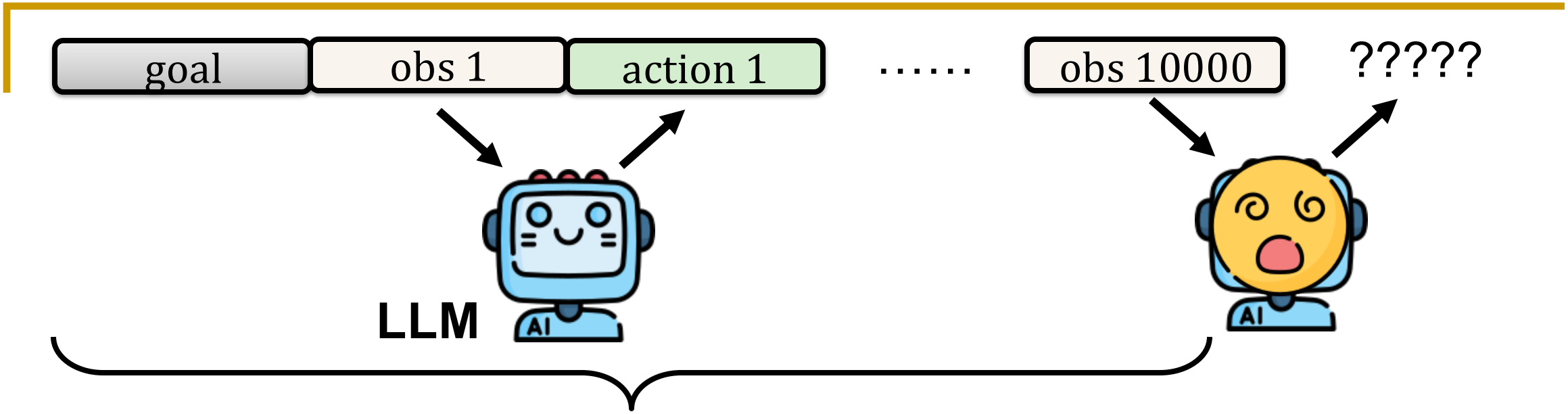
- Key directions for better AI Agents/ Agentic AI development for both academia and industry:



Adjust Behavior based on Experience



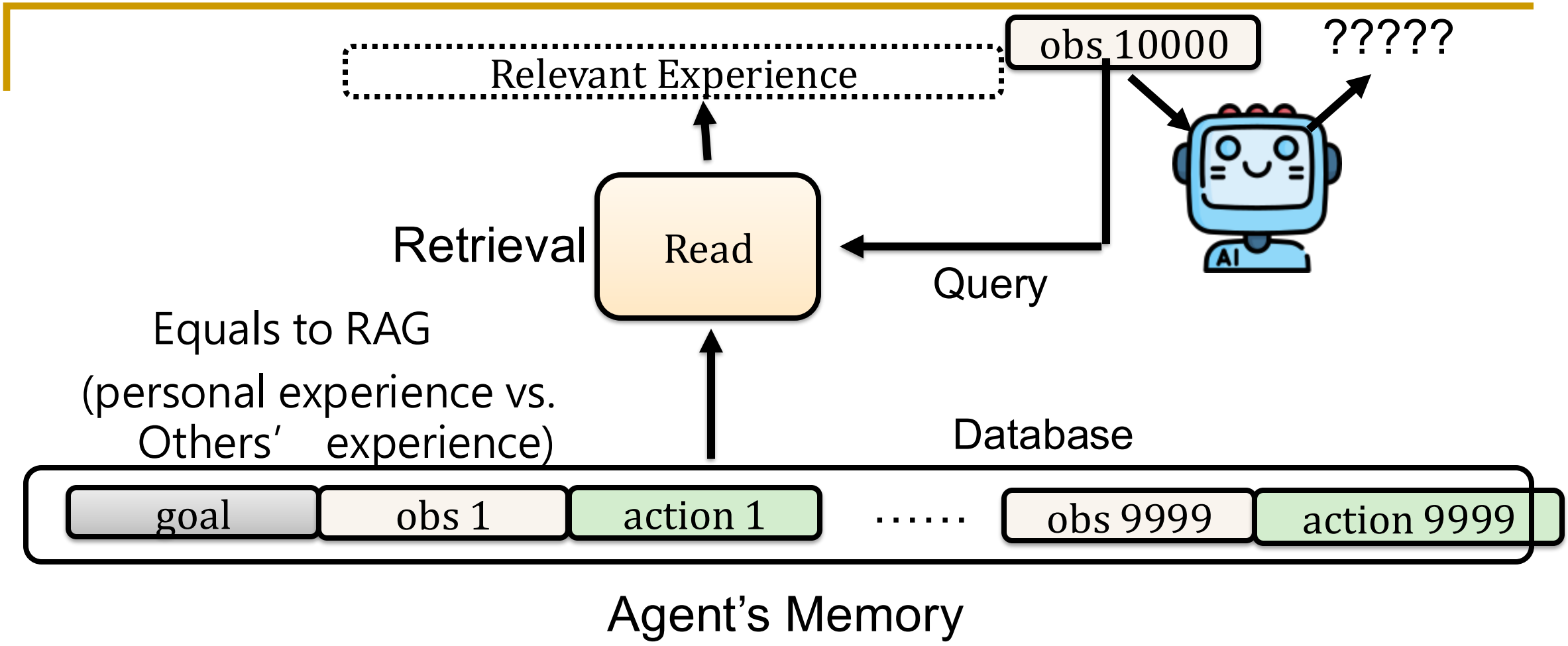
Adjust Behavior based on Experience



Reliving the entire Agent's life experiences...

Momorizing ability

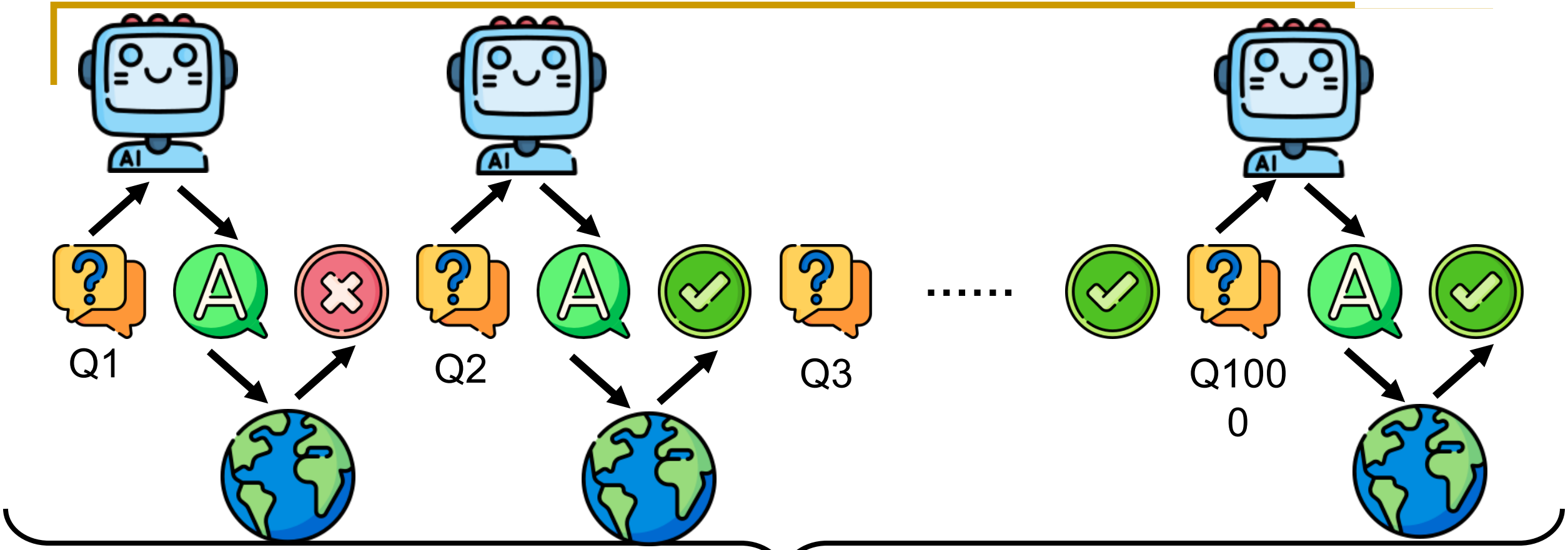
Adjust Behavior based on Experience



StreamBench

https://stream-bench.github.io/
(done by Appier Researchers)





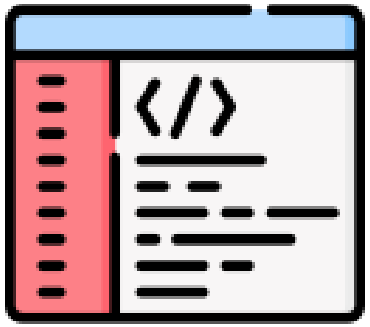
Goal: Maximize the accuracy over the sequence

Tool-Calling

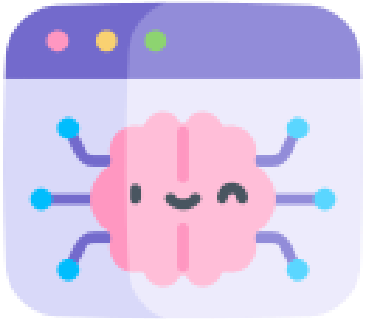
Tools: You only need to know how to use them, not how they work internally



Search Engine



Python



Other AI
(Different capabilities, stronger but costly)

- Tools can be viewed as functions, and using tools is equivalent to invoking these functions.

How to use tools

System Prompt

If you encounter a question you cannot answer based on your knowledge, use a tool.
Place the tool's command between <tool> and </tool>. After using the tool, you will receive output. Place this output between <output> and </output>.

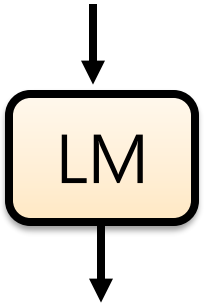
How to Use All Tools

The tools you can use now are as follows: The function Temperature(location, time) retrieves the temperature at a specific location and time.
Example usage: Temperature('Hong Kong, '2025.02.22 14:26')

How to use a specific tool

User Prompt

What was the temperature in Hung Hum at 2:00 PM on March 10, 2025?

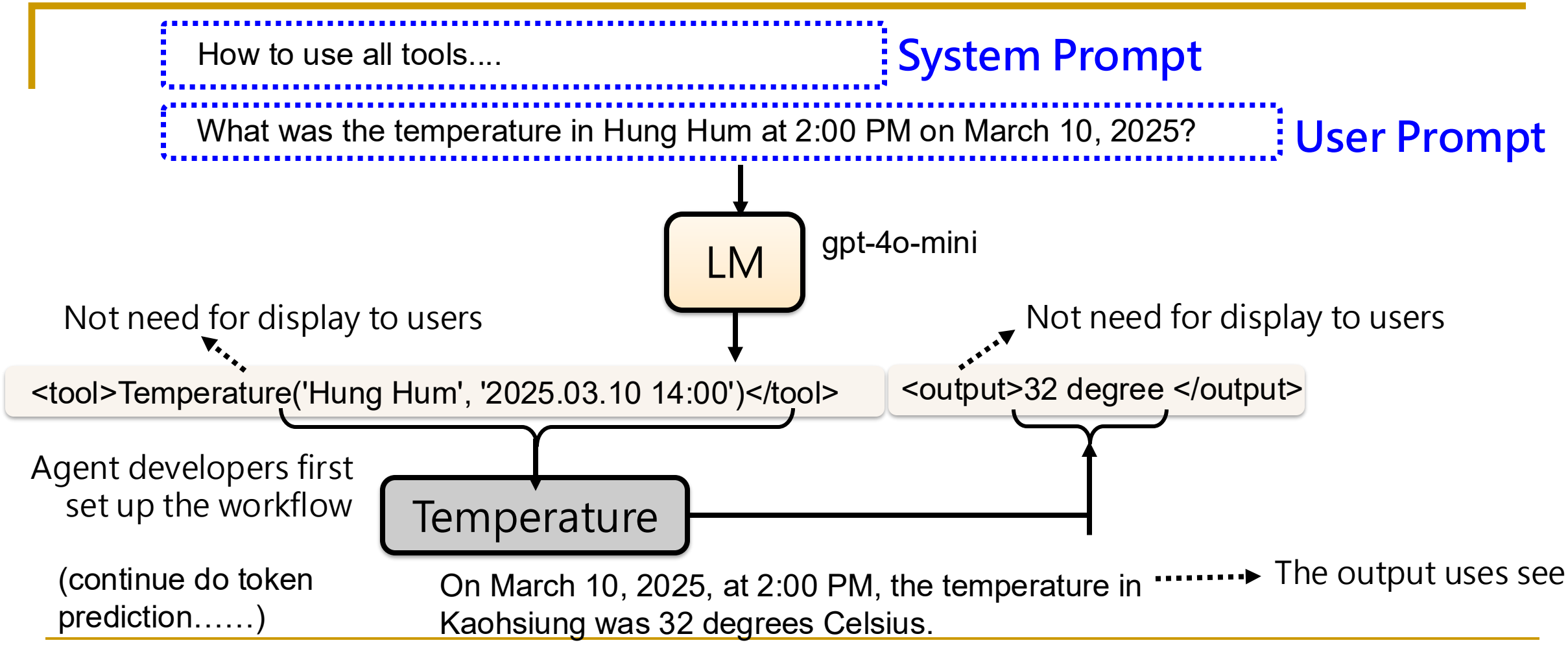


gpt-4o-mini

<tool>Tempature('Hung Hum', '2025.03.10 14:00')</tool>

This is just a string of text; it cannot actually call a function.

How to use tools



Most frequently used tool: Search engine

Retrieval Augmented Generation (RAG)



Using other AI as tools Using other AI



text query → LM → text response



What this people Say? → LM → She said: hello!



How’ s emotion? → LM → In a good mood



Modules
Speech Recognition
Language Identification
Speech Detection
Speech Emotion Recognition
Speech-to-Noise Ratio (SNR) Estimation
Reverberation Detection
Accent Classification
Stress Position Identification
Spoofing Detection
Music Chord Classification
Sythetic Speech Detection
Speaker Verification
Speaker Diarization
Sound Classification
Query LLM
Speaker Distance Estimation

Could you take a look at a voice message we received? Is it Sarah? What she said? Any clues in the background about where she is? Does she seem upset? I am worried about her.

Yes, the speaker verification result confirmed that it is Sarah.

Sarah said: "Hi, it's Sarah. I'm calling to let you know I'll be late to the meeting. I'll be there in about 30 minutes. Sorry for the inconvenience."

The background noise includes traffic sounds, suggesting that she might be outdoors near a road or in a vehicle.

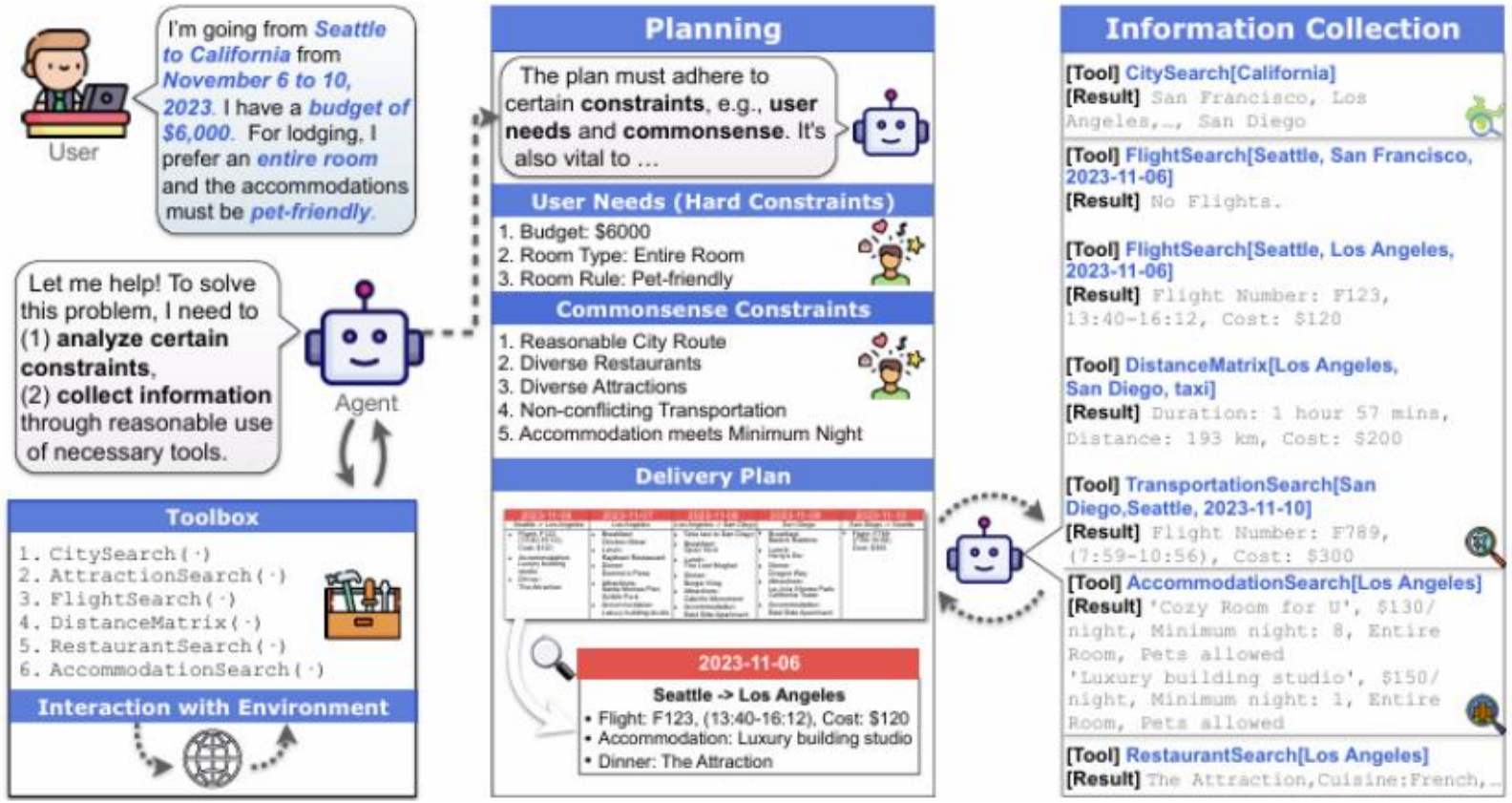
The recognized emotion is sad. Combined with the context of her message (apologizing for being late), it suggests she might be feeling a bit upset or stressed about the delay.

https://arxiv.org/abs/2407.09886

Planning

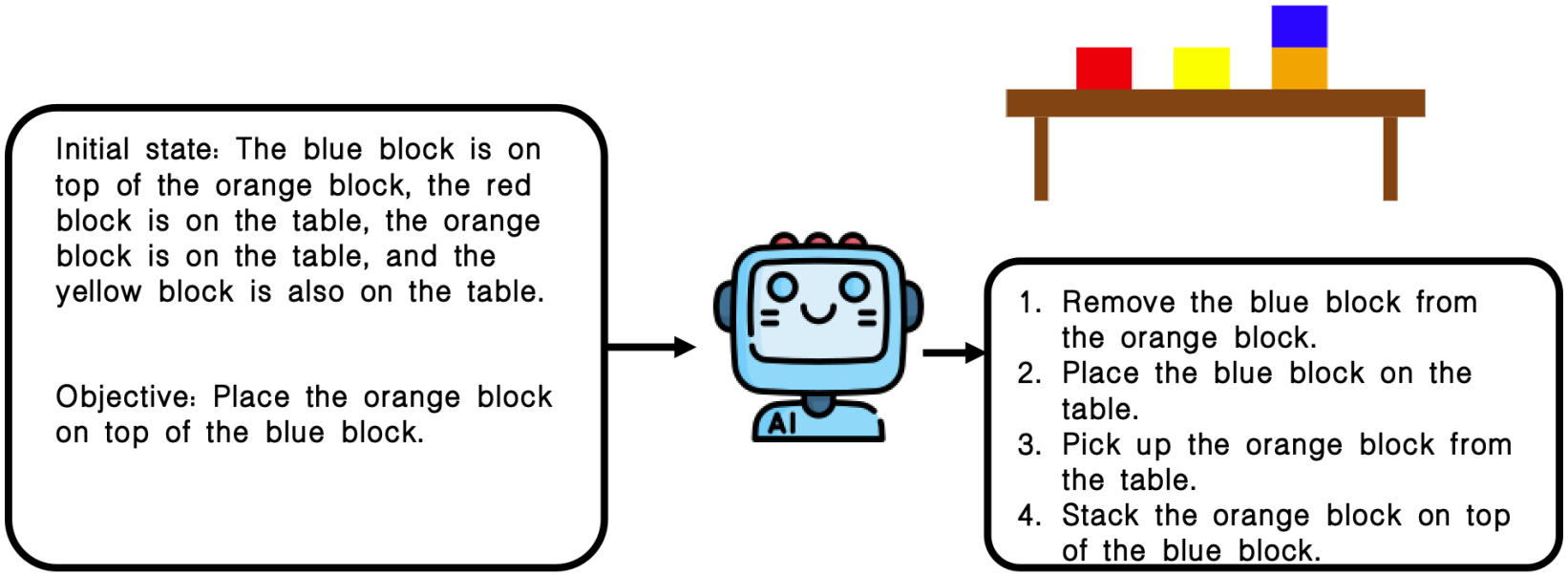
- Complex agents can't act without making a decision, and they can't make good decisions without first making a plan. Agentic planning consists of several key components that work together to encourage optimal decision-making.
  - Goal definition
  - State representation
  - Action sequencing
  - Optimization and evaluation
  - Collaboration

TravelPlanner



https://arxiv.org/abs/2402.01622

Things to Think…



Could it be that the LLM has already "know" (memorize) such knowledge