

南京大学
NANJING UNIVERSITY

AI Agent
From Neuro-Symbolic Perspective

郭兰哲

南京大学

https://www.lamda.nju.edu.cn/guolz/

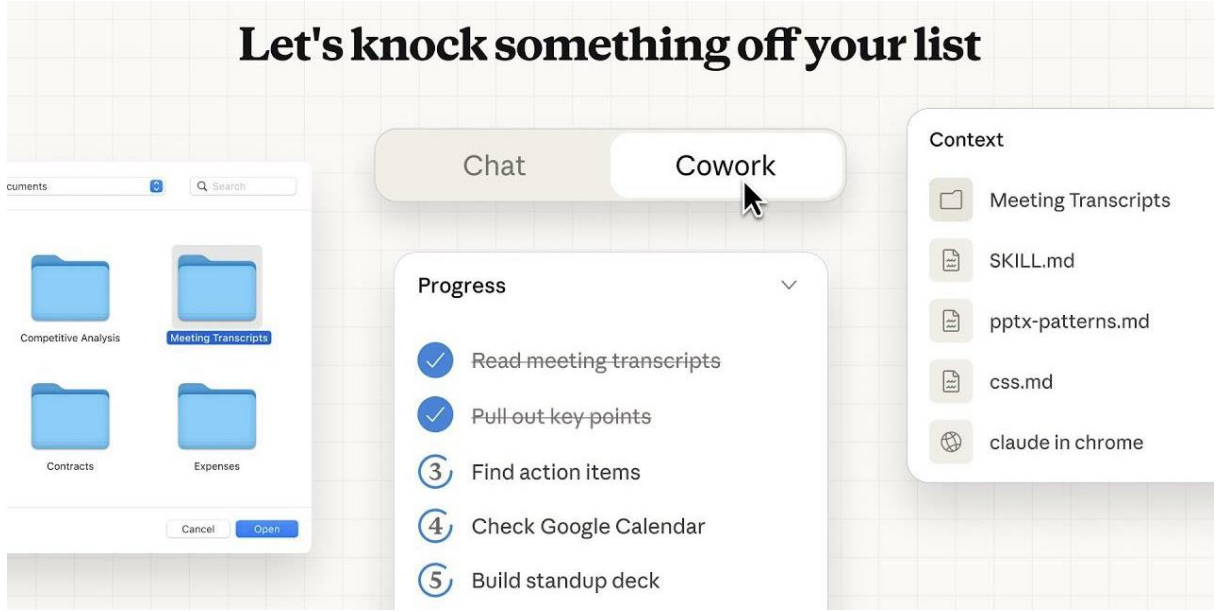
What is an Agent?

Gemini

CURSOR

OpenAI Codex

Claude

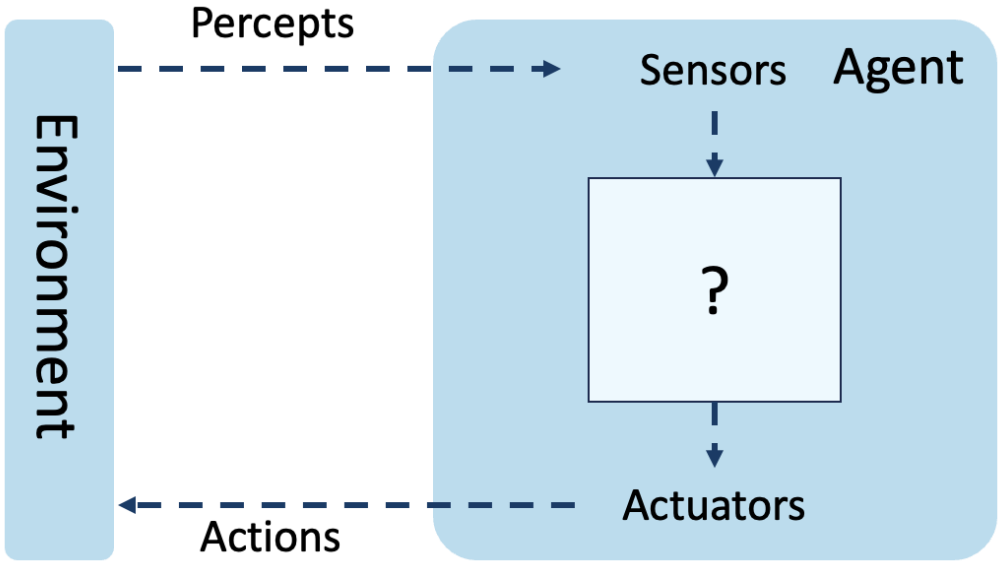








What is an Agent?



An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators.

— Russell & Norvig, Artificial Intelligence: A Modern Approach

- **Environment**: a repository, website, game or physical world
- **State/Observations**: file contents, webpages, tool results, screenshots
- **Actions**: replies, file edits, shell commands, API calls, and mouse or keyboard events
- **Reward**: passing test cases, satisfying LLM-as-a-judge rubrics, positive user feedback

Two Views of Agent

LLM-first view

- Idea: 让我们把LLM变成Agent
- 关键：如何在LLM的基础上一步步升级，从Chatbot到Reasoner再到Agent？

Agent-first view

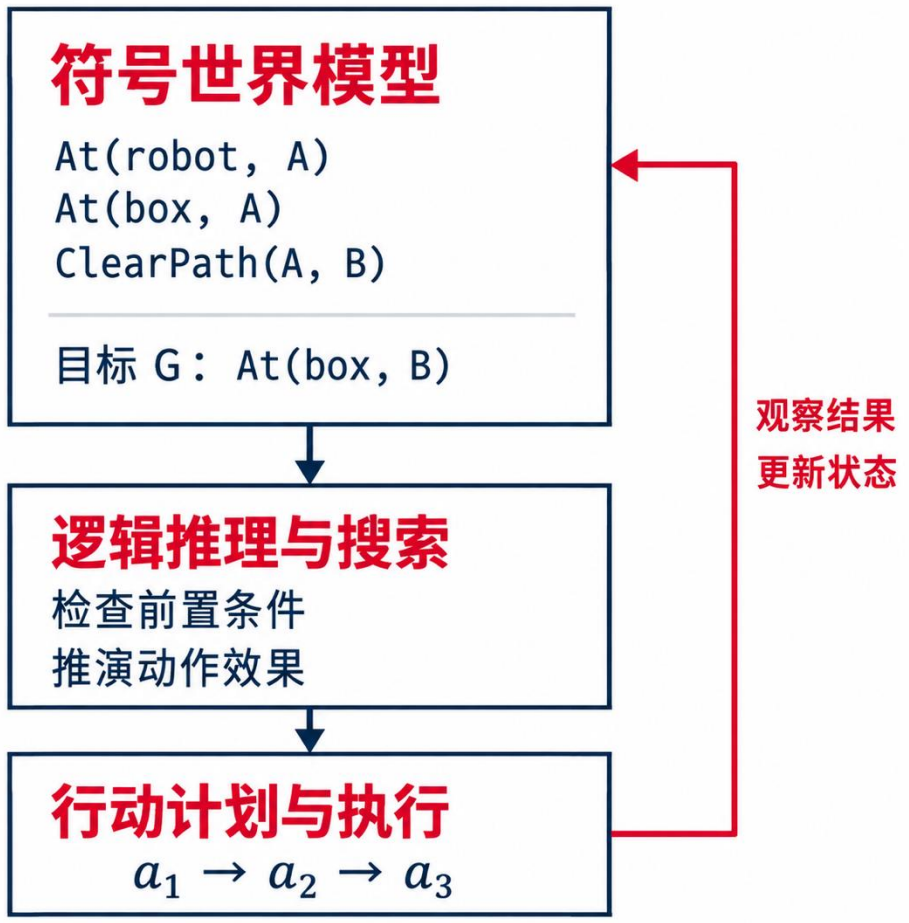
- Idea: 让我们把LLM放到Agent架构中
- 关键：Agent本身有哪些挑战？LLM的引入可以帮助缓解哪些挑战，还存在哪些新的挑战？

Evolution of AI Agents: Symbolic Agent

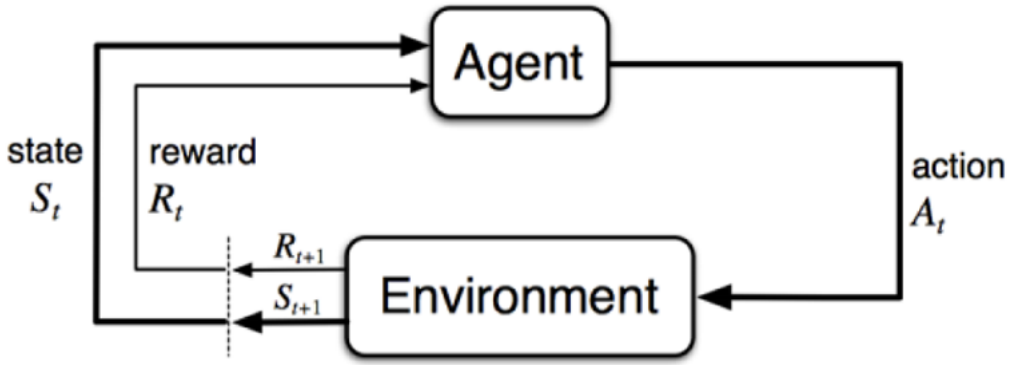
Shakey the Robot (1969)

第一个能够推理自身行为，并与物理世界交互的通用移动机器人，能够自主规划路径和动作完成“推箱子”任务



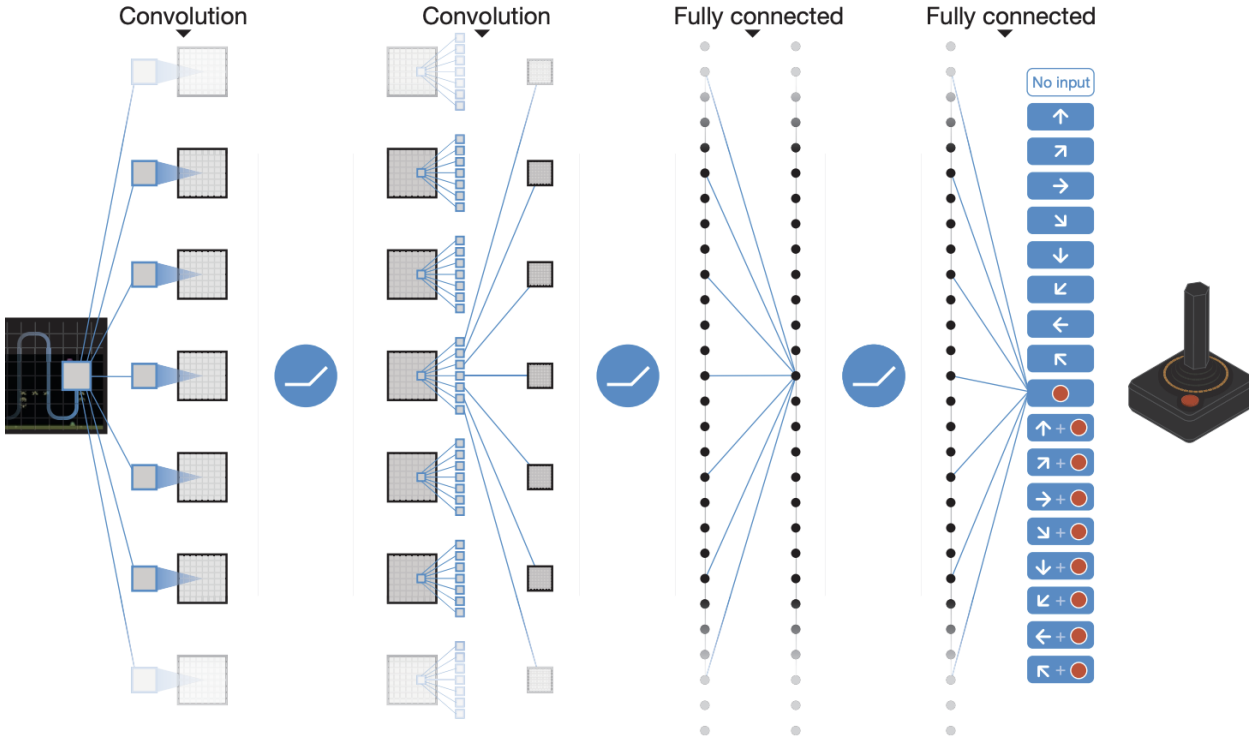


Evolution of AI Agents: RL Agent



Agent-environment loop

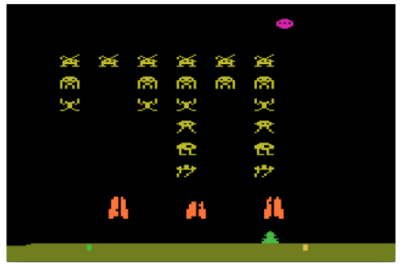
(Sutton and Barto, 1998)



DQN on Atari (Mnih et al., 2015)



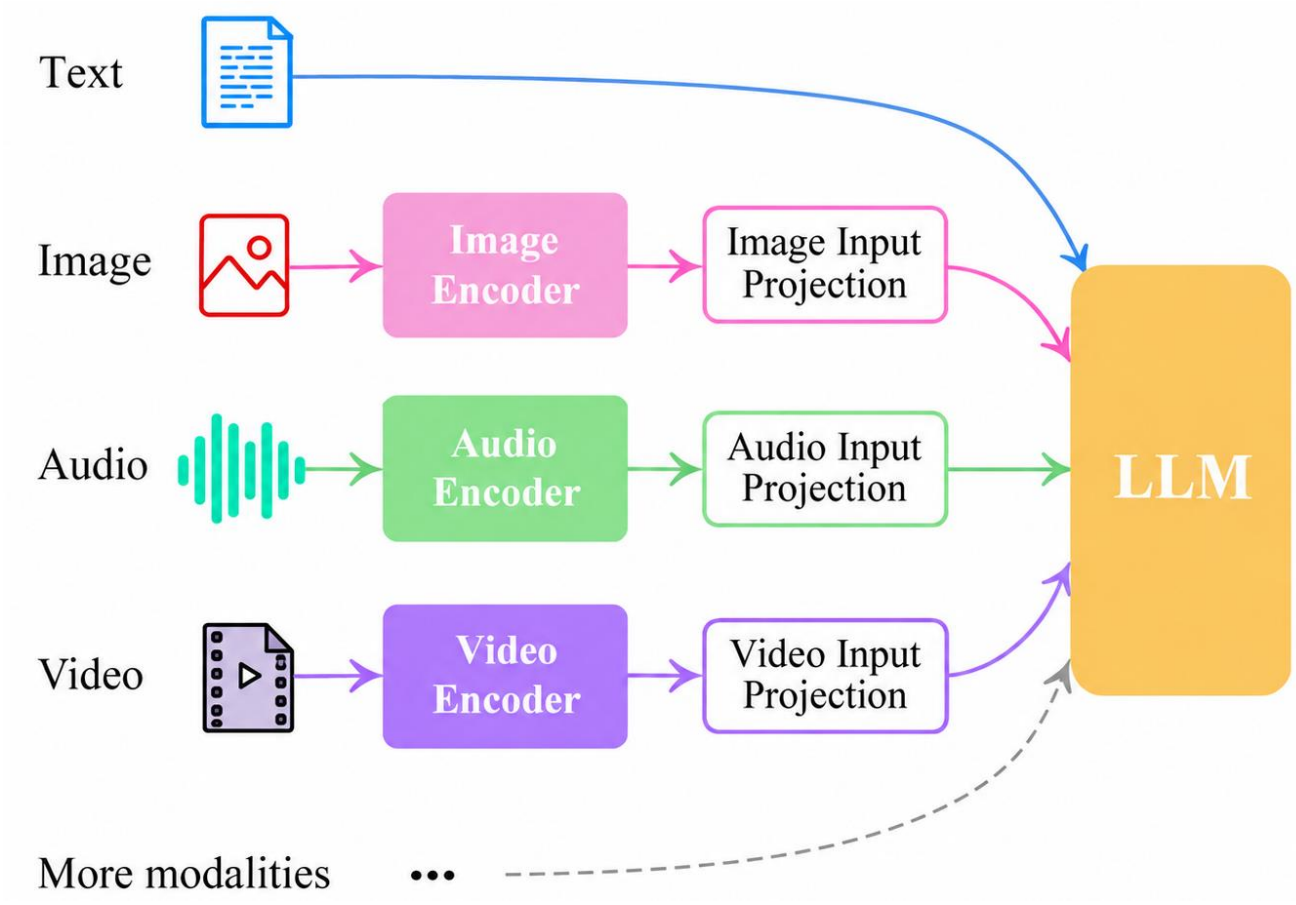








What LLMs Made Possible: General Perception and Reasoning



Chain of Thought Prompting

Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

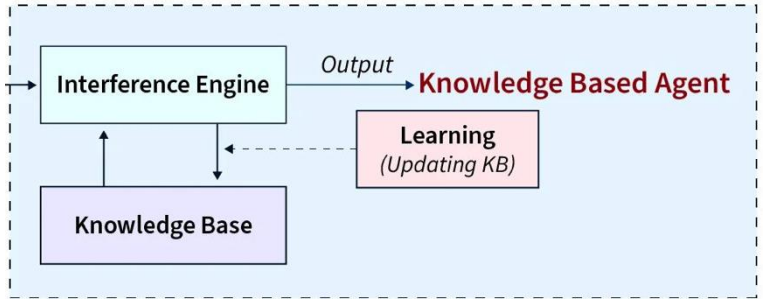
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

Evolution of AI Agents: Agent-First View

Symbolic Agent

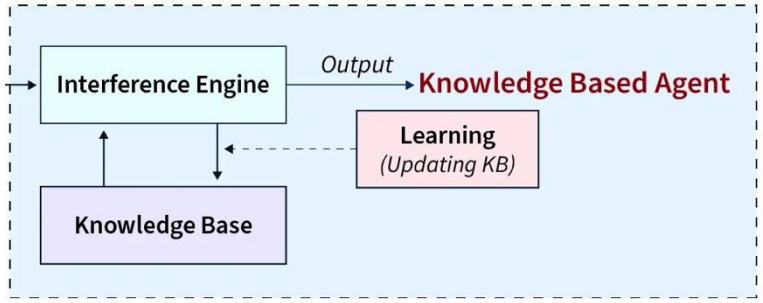


显式符号规则+推理规划算法

感知能力	缺乏符号接地能力		
推理能力	严谨符号逻辑推理		
适应能力	依赖预先定义的符号空间		

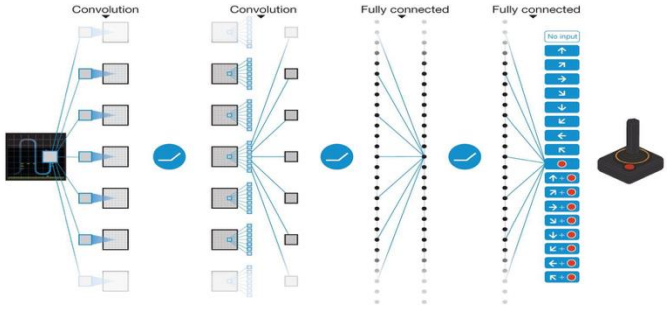
Evolution of AI Agents: Agent-First View

Symbolic Agent



显式符号规则+推理规划算法

Neural & RL Agent

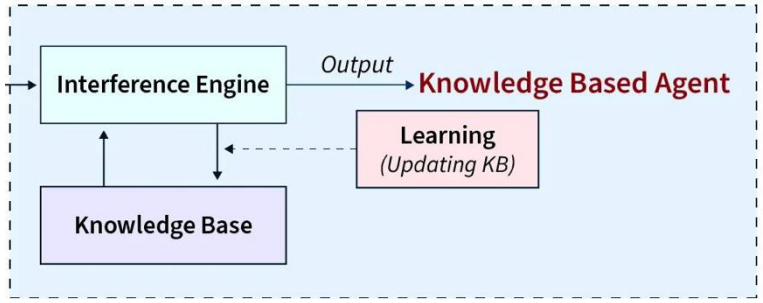


神经网络感知+强化学习策略

感知能力	缺乏符号接地能力	神经网络的感知能力	
推理能力	严谨符号逻辑推理	强化学习策略	
适应能力	依赖预先定义的符号空间	样本效率低，难以在不同场景迁移	

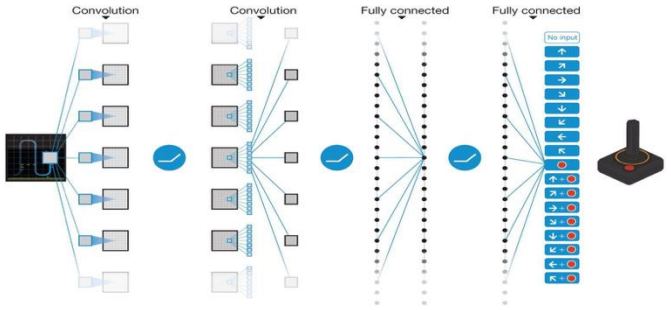
Evolution of AI Agents: Agent-First View

Symbolic Agent



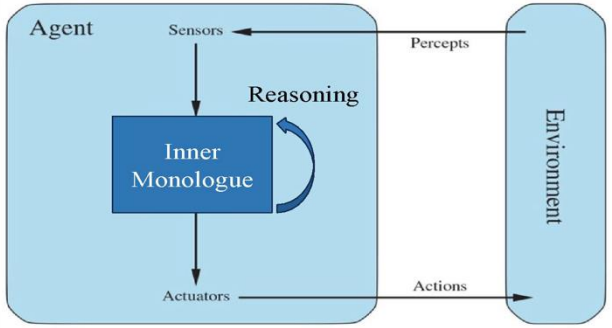
显式符号规则+推理规划算法

Neural & RL Agent



神经网络感知+强化学习策略

LLM Agent



LLM通用感知与推理

感知能力	缺乏符号接地能力	神经网络的感知能力	高，几乎任意模态数据
推理能力	严谨符号逻辑推理	强化学习策略	基于语言的思维链推理
适应能力	依赖预先定义的符号空间	样本效率低，难以在不同场景迁移	大模型自身的通用性

Is LLM All We Need?

Every weekday at 9am, an AI agent wakes up, drafts fixes, and opens pull requests before anyone sits down

One morning it reads a strange error and decides the fix is to **delete the test folder**

Then it confidently reports:

“Done! All tests pass.”

Nothing stopped it.

Nothing checked it.

Nothing wrote it down.

Same model as yesterday. Same prompt as yesterday.

Beyond LLM, What We need?



The harness is everything *around* the model: what it may do, what it knows, how its work is proven, what happens when it goes wrong.

Anatomy of Agent Harness

1 · Tools & Action System

The actions the model can take and the shape of each one.

2 · Memory & Context

What the model sees: what goes in, what gets trimmed, what lives in files.

3 · Agentic Loop

Agent observes, plans, acts, and uses feedback to revise its plan until a goal is reached.

4 · Safety & Permissions

The parts that say no: permissions, limits, checks.

Tools & Action System

Tools & Action System

- Just like humans, LLMs suffer from the similar limitations. But in the same way,
- LLMs + tool use
  - Enhanced scalability
  - Improved consistency
  - Greater interpretability
  - Higher capacity and productivity











matplotlib

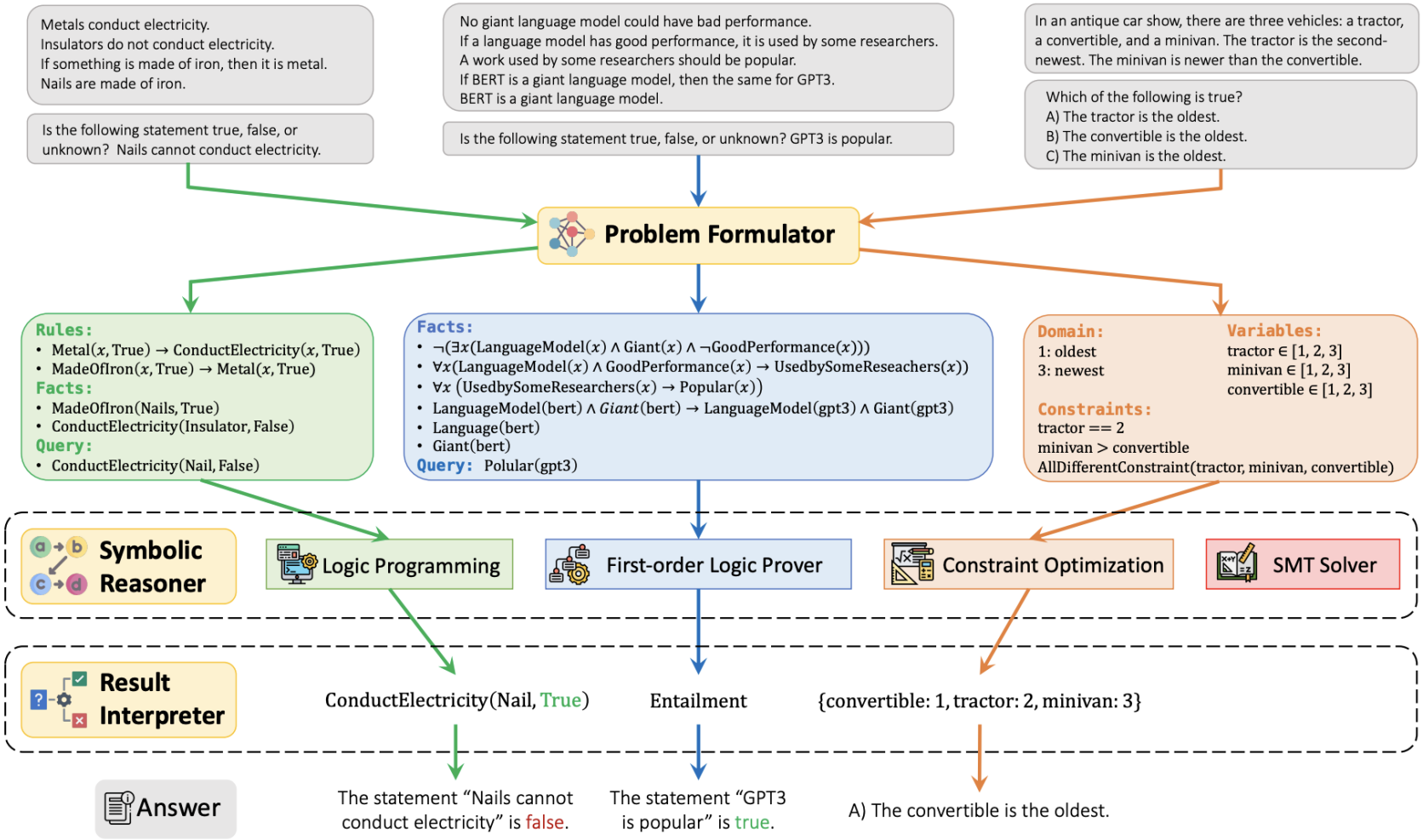
Search Google or type a URL

Symbolic Solver as Tools

Convert **Natural Language** into **Formal Language**, then call a **symbolic solver**, such as

- First-order Logic Prover
- SMT Solver
- Constraint Optimizer

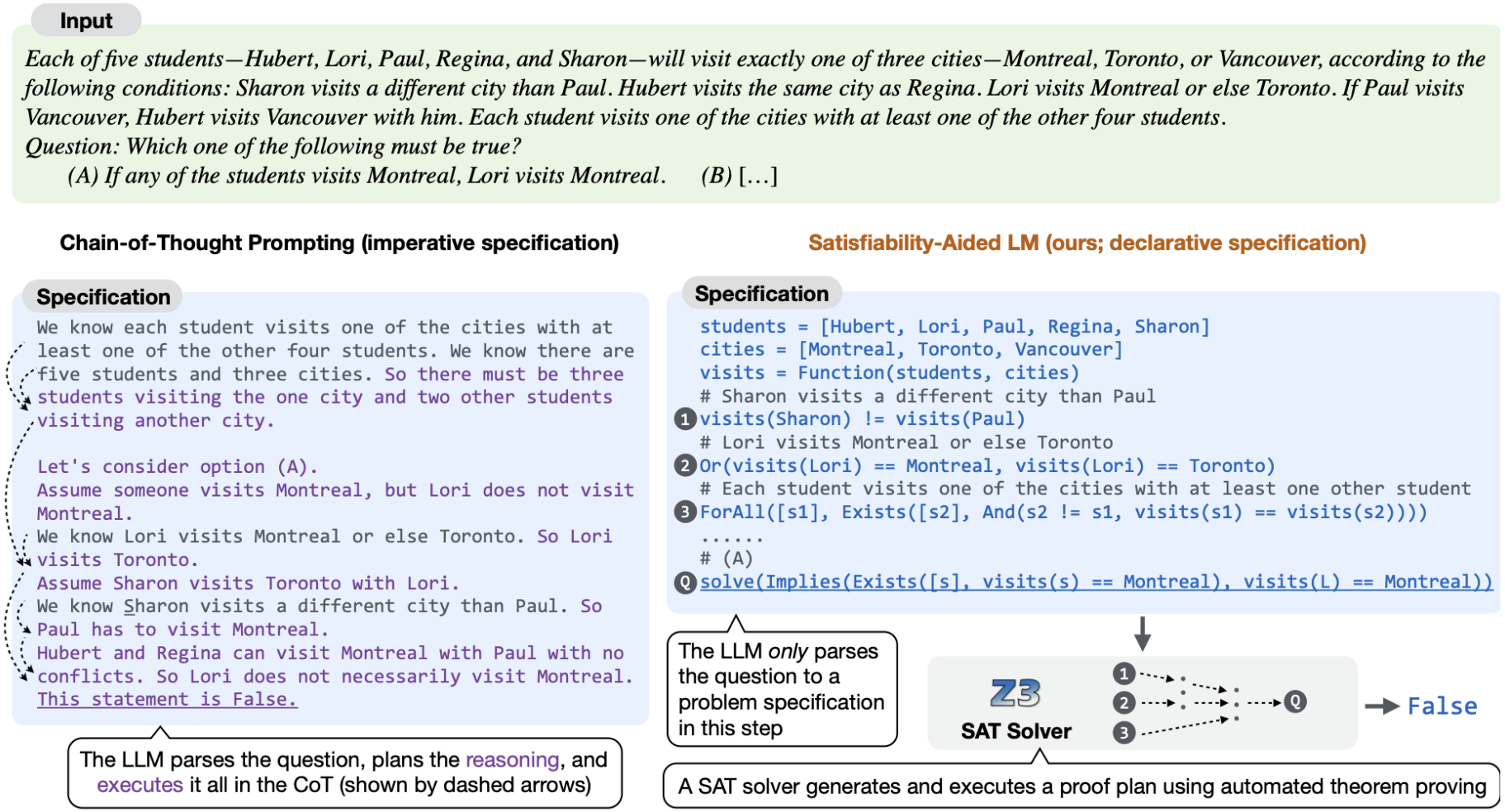
…



[1] LOGIC-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning. Pan et al., EMNLP 2023.

Symbolic Solver as Tools

Parse an **natural language** input into a declarative task specification (a set of logical constraints) , then use a SAT solver (e.g., Z3) to solve the problem

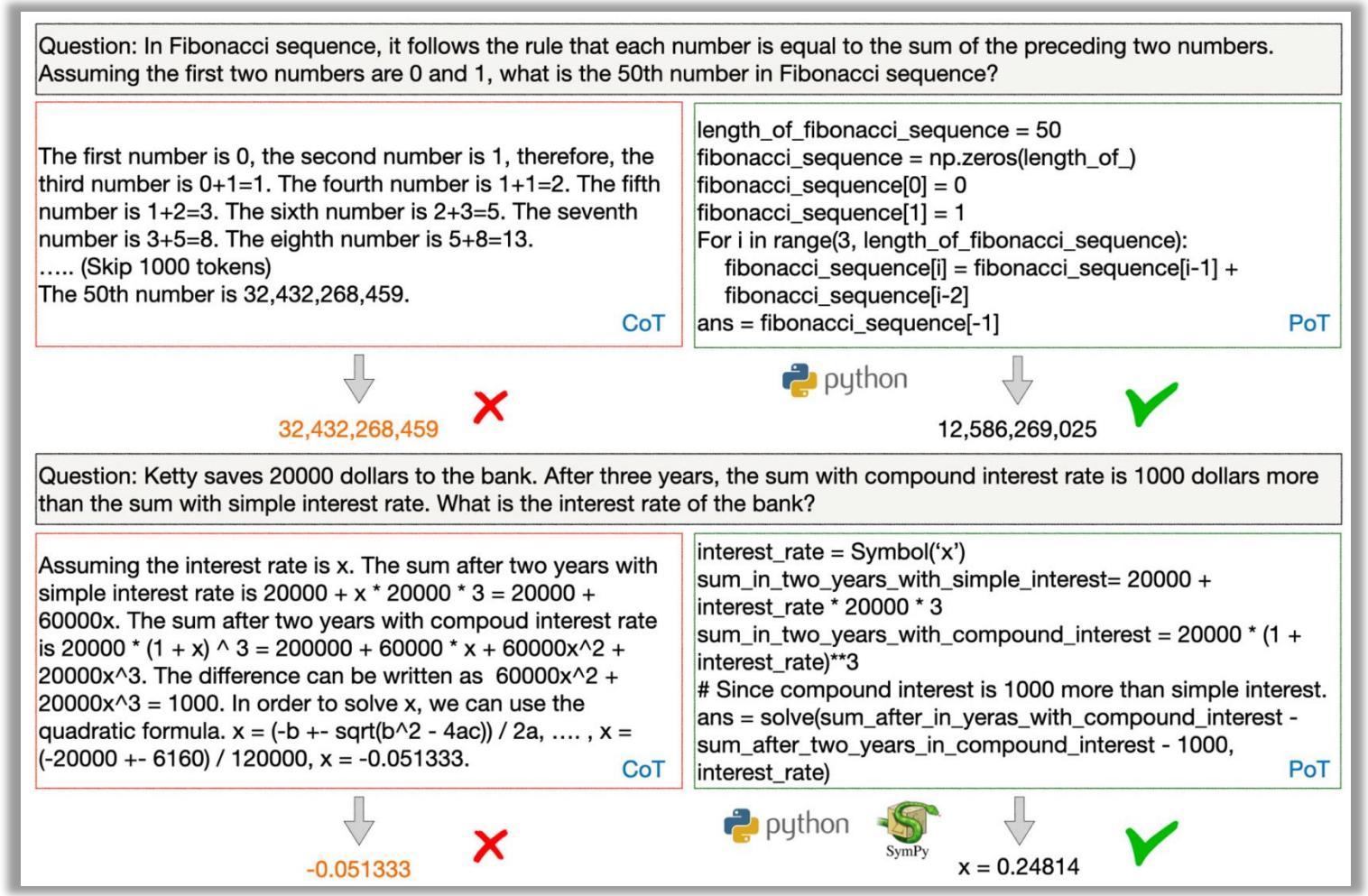


[1] SATLM: Satisfiability-Aided Language Models Using Declarative Prompting. Ye et al., NeurIPS 2023.

Program Augmented Reasoning

Program of Thought

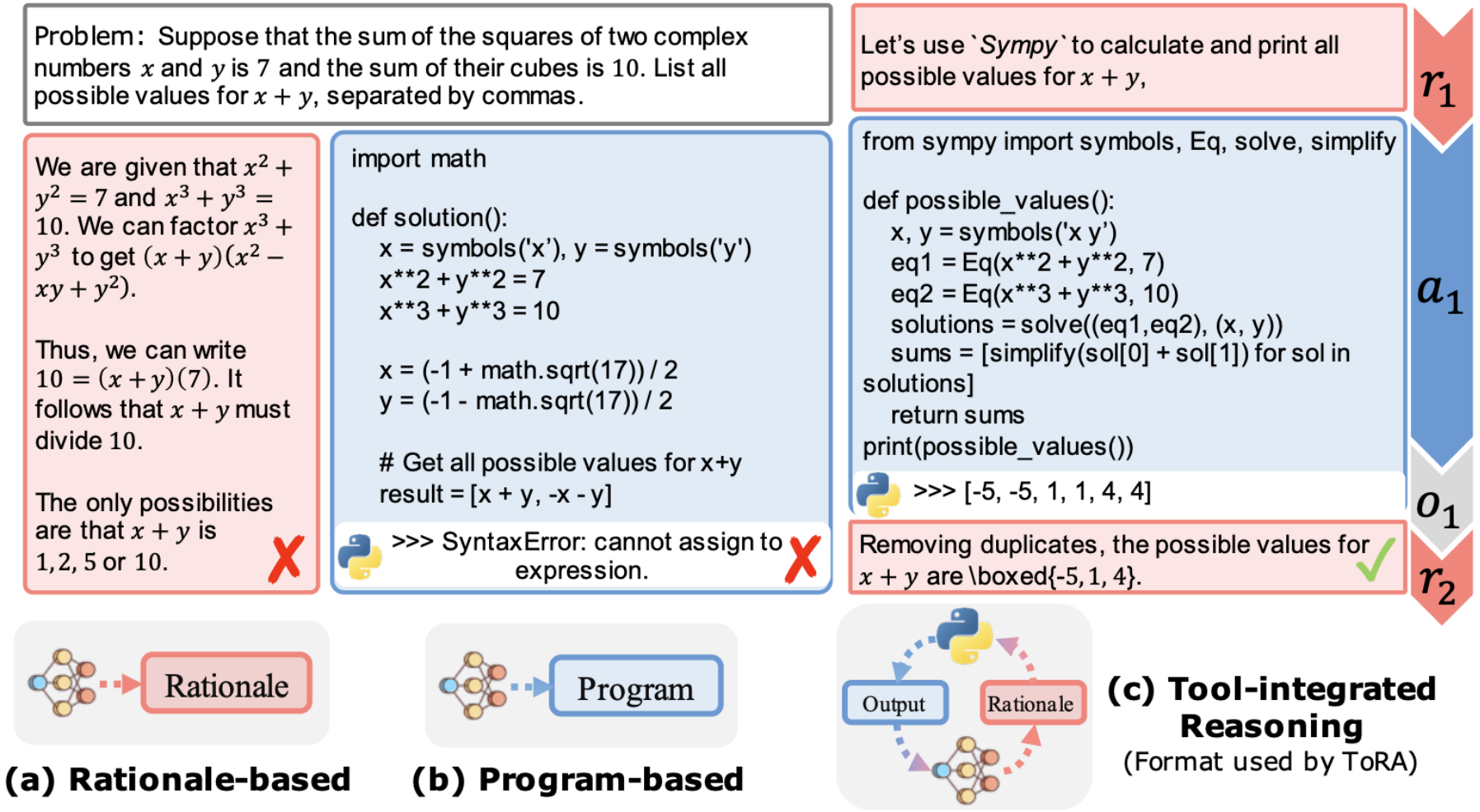
Convert the reasoning process expressed in natural language into a **programming language** (e.g., Python, etc.), and then **execute the program interpreter**



[1] PAL: Program-aided Language Models. Gao et al., ICML 2023.
[2] Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks. Chen et al., TMLR 2023.

Program Augmented Reasoning

Integrate natural language reasoning with the utilization of external tools (e.g., computation libraries, symbolic solvers)



[1] ToRA: A Tool-Integrated Reasoning Agent for Mathematical Problem Solving. Gou et al., ICLR 2024.

Program Augmented Reasoning

Multi-Modal Reasoning

LLMs generate code to manipulate images for visual reasoning




```
# Define the cropping coordinates (x1, y1, x2, y2)
x1, y1, x2, y2 = 1750, 1950, 1850, 2600

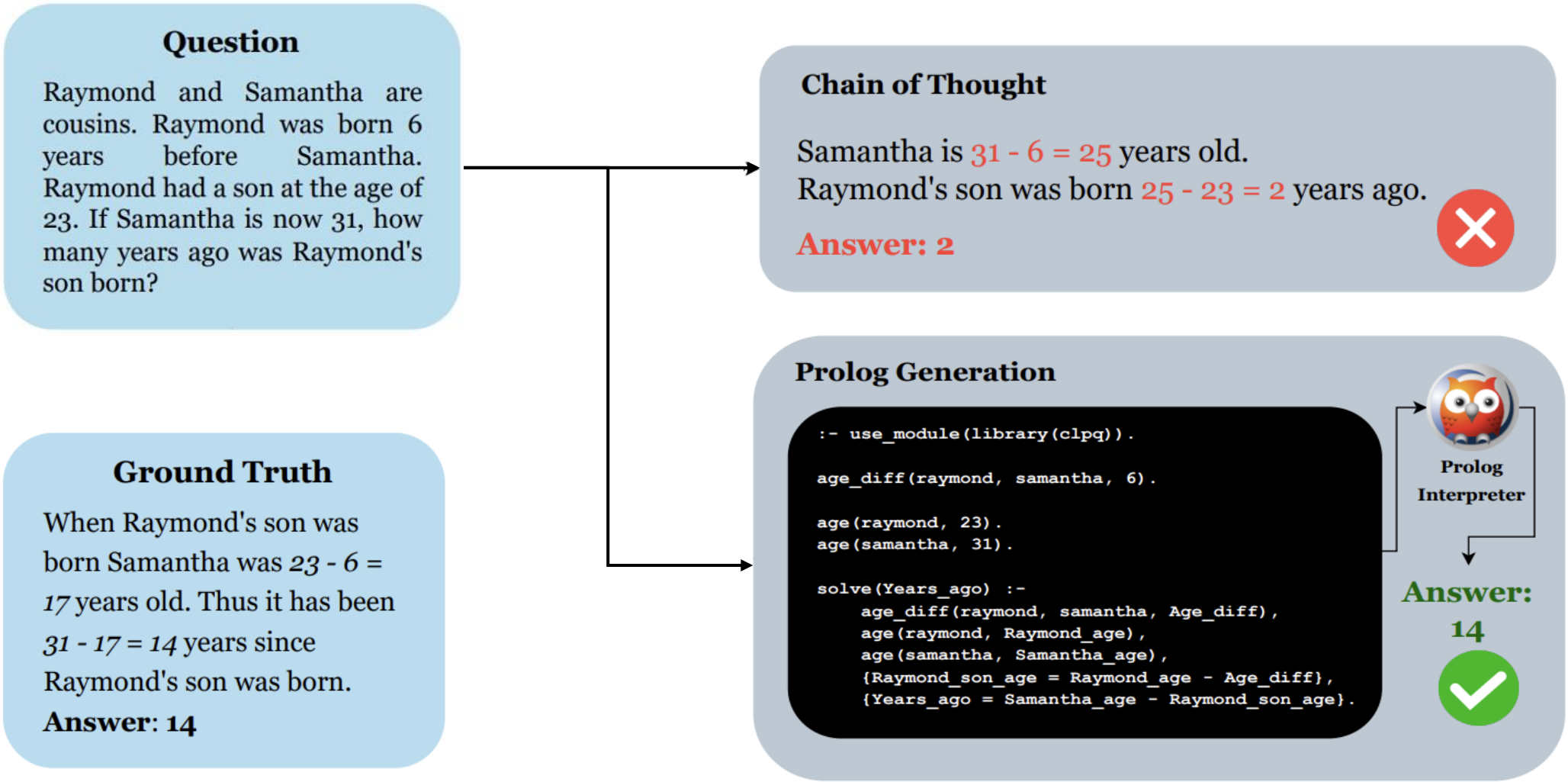
# Crop the image
cropped_image = image[y1:y2, x1:x2]

# Apply zoom (resize the cropped image to enhance details)
zoom_factor = 2
zoomed_image = cv2.resize(cropped_image, None,
fx=zoom_factor, fy=zoom_factor,
interpolation=cv2.INTER_CUBIC)

# Save the processed image with a random suffix
random_suffix = random.randint(1000, 9999)
processed_filename = f"224_{random_suffix}.jpg"
processed_path = "/mnt/data/temp_processed_images/" +
processed_filename
cv2.imwrite(processed_path, zoomed_image)
```

[1] Thyme: Think Beyond Images. Zhang et al., ICLR 2026.

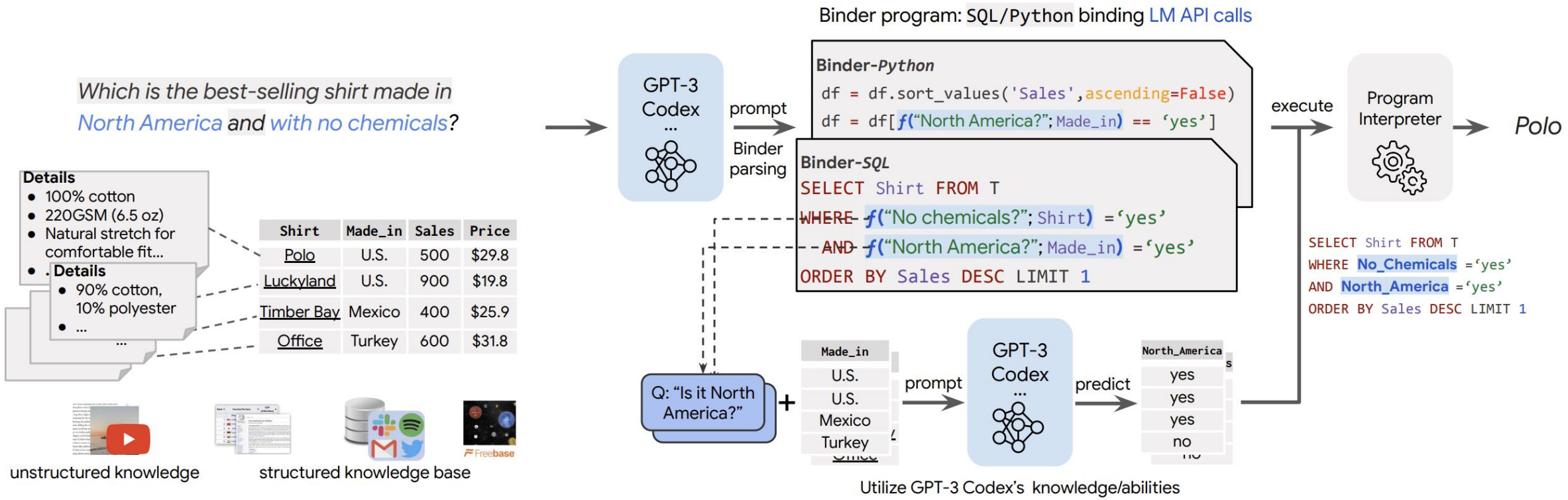
Program Augmented Reasoning



[1] Arithmetic Reasoning with LLM: Prolog Generation & Permutation. Yang et al., ACL 2024.

Program Augmented Reasoning

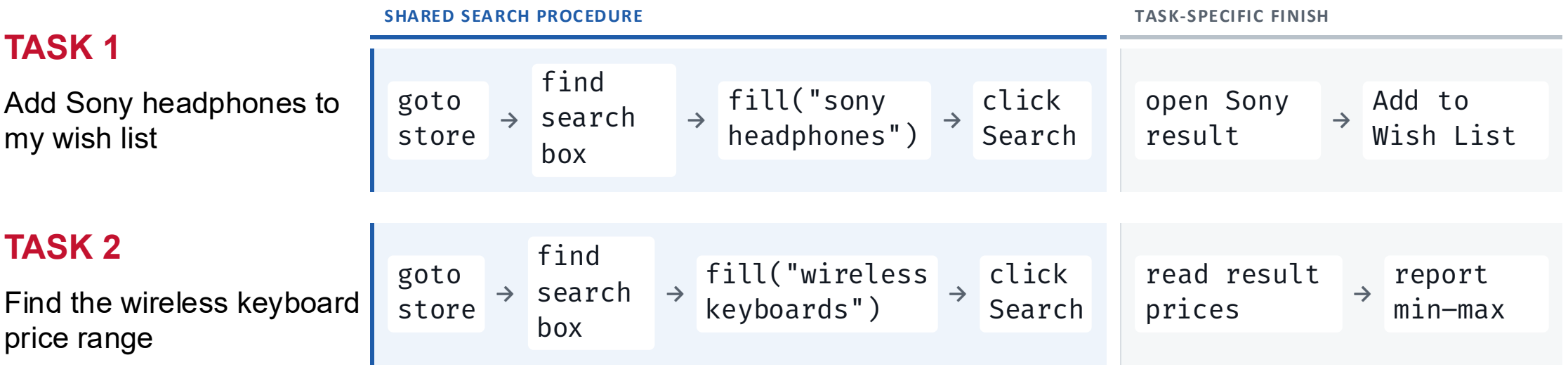
- **Parse Stage**: LLM maps input to **Python/SQL** algorithm
- **Execution Stage**: A program interpreter executes the program to derive the answer



[1] Binding Language Models in Symbolic Languages. Cheng et al., ICLR 2023.

Memory & Skills

Shared Task Structure



Lots of structure recurs in agent tasks

Type of Experience

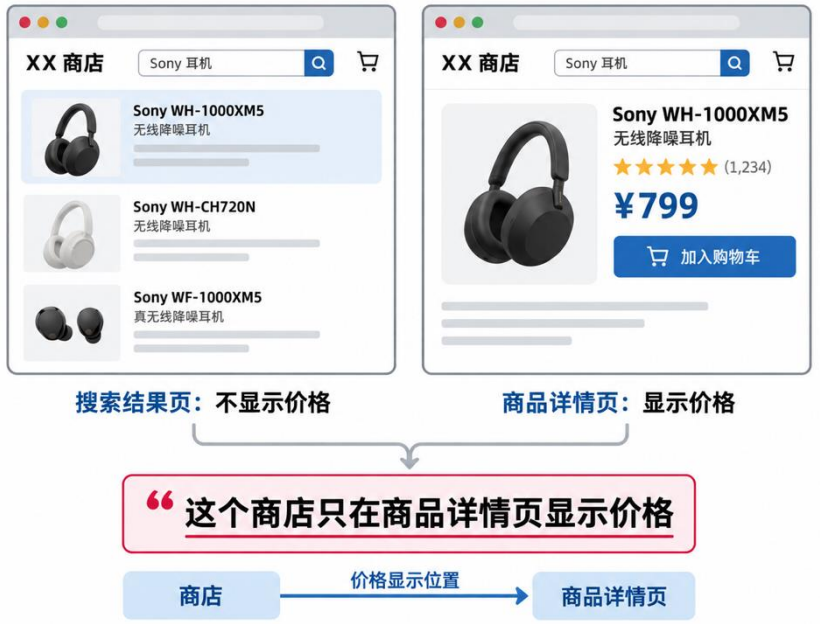
Episode

A full trace, every observation and action



Fact

"This store shows prices only on the product page"

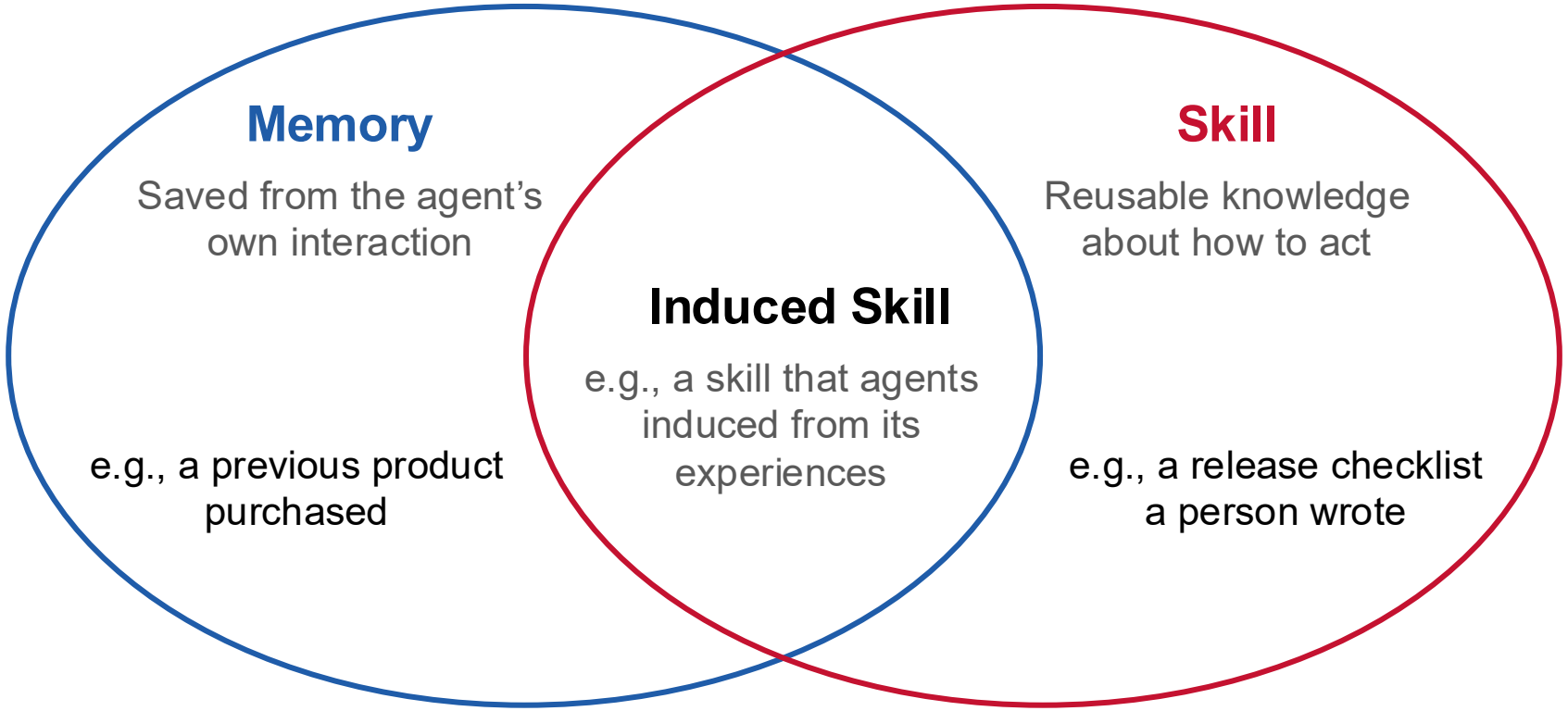


Skill

Fill the search box -> click search -> open results



Memory vs Skill



Skills as Reusable chunks of a task

Search for a Product

Show me results for {query}

```
# To find {query}, I will type it into the store's search box and submit.
fill('element145', {query})
click('element147')
```

Reuse →

Add steps ↘

Report a Price Range

What is the price range for {query}?

```
# To get the price range for {query}, I will first search the store for it.
fill('element145', {query})
click('element147')
```

```
# The results list a price on each card. I will read the lowest and the highest.
send_msg_to_user("{low} to {high}")
```

Shared Task Structure

1 · Representation

可复用的知识以什么形式存在？

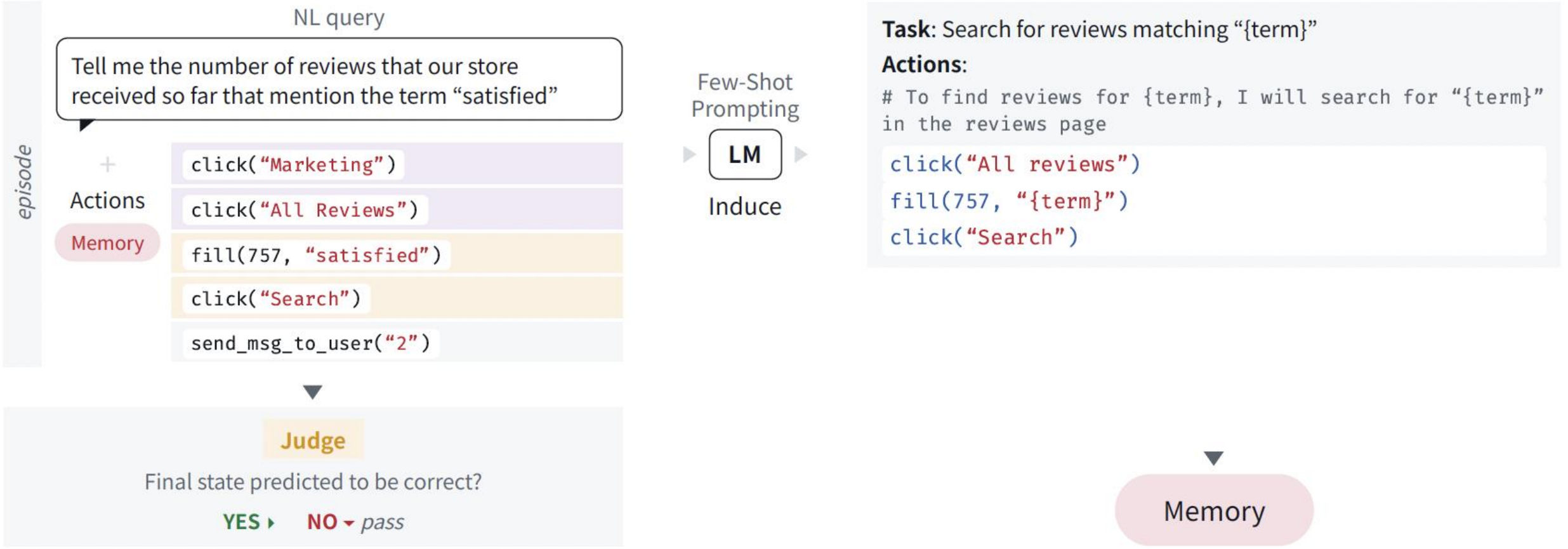
2 · Induction

如何从经验中自动学到可复用的技能？

3 · Adaptation

如何持续更新技能，适应环境？

Inducing textual representations of Skill



[1] Agent Workflow Memory. Wang et al., ICLR 2025.

An Induction Prompt

INDUCTION PROMPT FROM AGENT WORKFLOW MEMORY

```
Given a list of web navigation tasks, your task is to extract the common workflows.

Each given task contains a natural language instruction, and a series of actions
to solve the task. You need to find the repetitive subset of actions across
multiple tasks, and extract each of them out as a workflow.

Each workflow should be a commonly reused sub-routine of the tasks. Do not
generate similar or overlapping workflows. Each workflow should have at least
two steps. Represent the non-fixed elements (input text, button strings) with
descriptive variable names as shown in the example.
```

[1] Agent Workflow Memory. Wang et al., ICLR 2025.

Skills as Text vs Skill as Symbolic

Text + Examples

Task: Search for gaming accessories within a date range

Action trajectory:

```
click(1274) # Navigate to the Video Games category
fill(473, {search_terms}) # Enter search terms including product name and year
click(478) # Execute the search
```

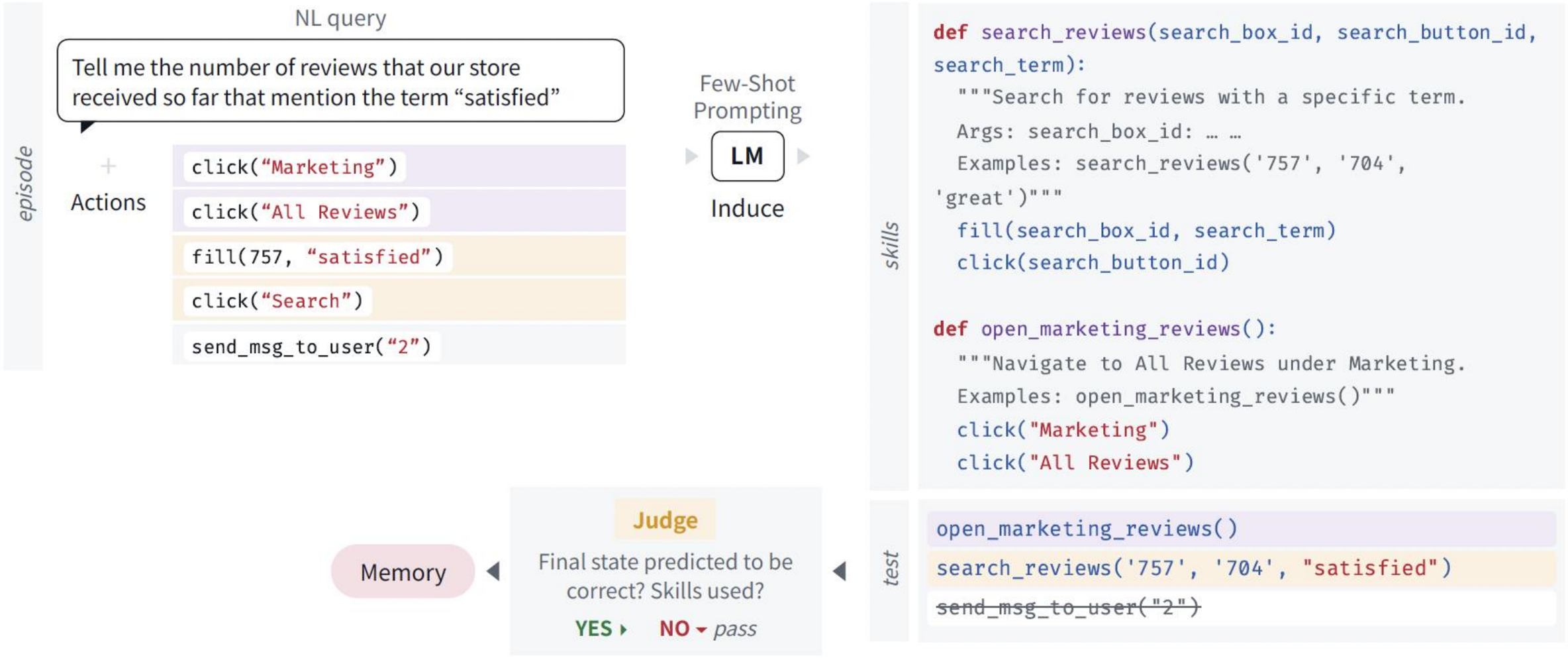
Symbolic Code

```python
def search_product(search_box_id: str, query: str):
    """Search for a product using the search box.
    Args:
        search_box_id: ID of the search input field
        query: Search query string to enter
    Returns:
        None
    Examples:
        search_product('595', 'sony bluetooth headphones')
    """
    click(search_box_id)
    fill(search_box_id, query)
    keyboard_press('Enter')
```

- **灵活性**：模型可以根据具体情境理解并变通
- **表达能力强**：适合描述模糊知识
- **不确定性**：同一指令多次运行难以保证一致
- **难以验证**：难判断模型是否真的按步骤做了
- **精确计算弱**：涉及数值计算容易出错

- **可测试、可验证**：能写单元测试，能定位错误位置
- **可组合、可复用**：函数模块容易组合、拼装
- **精确计算**：涉及解析、计算、API调用更准确
- **泛化性差**：难以表达模糊知识，难应对意外输入

Inducing Code Skills



[1] Inducing Programmatic Skills for Agentic Tasks. Wang et al., COLM 2025.

Code Skills allow Testing

CODE SKILL ADMISSION

```
candidates = propose_skill(task)
episode = practice(candidates, task)

if reward_model(episode.actions, episode.screenshots, task):
    memory.add(candidate)
else:
    candidate = revise(candidate, task)
```

EXAMPLE OF LINTING AND TESTING A CODE SKILL

```
identify_pill(page, imprint="M366", color="White")
```

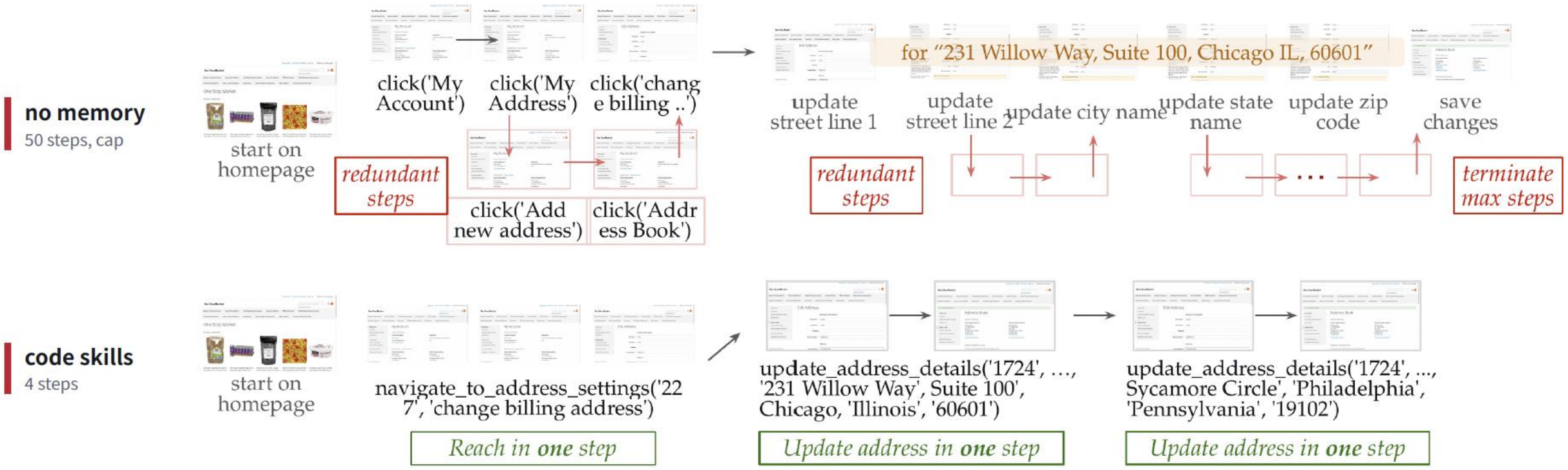
```
WARNING  parameter 'color' is defined but never used in the function body
```

```
+ if color:
+     await page.get_by_role("group", name="Color and shape (optional)") \
+                 .get_by_role("combobox", name="Color (optional)").select_option(color)
```

[1] Inducing Programmatic Skills for Agentic Tasks. Wang et al., COLM 2025.

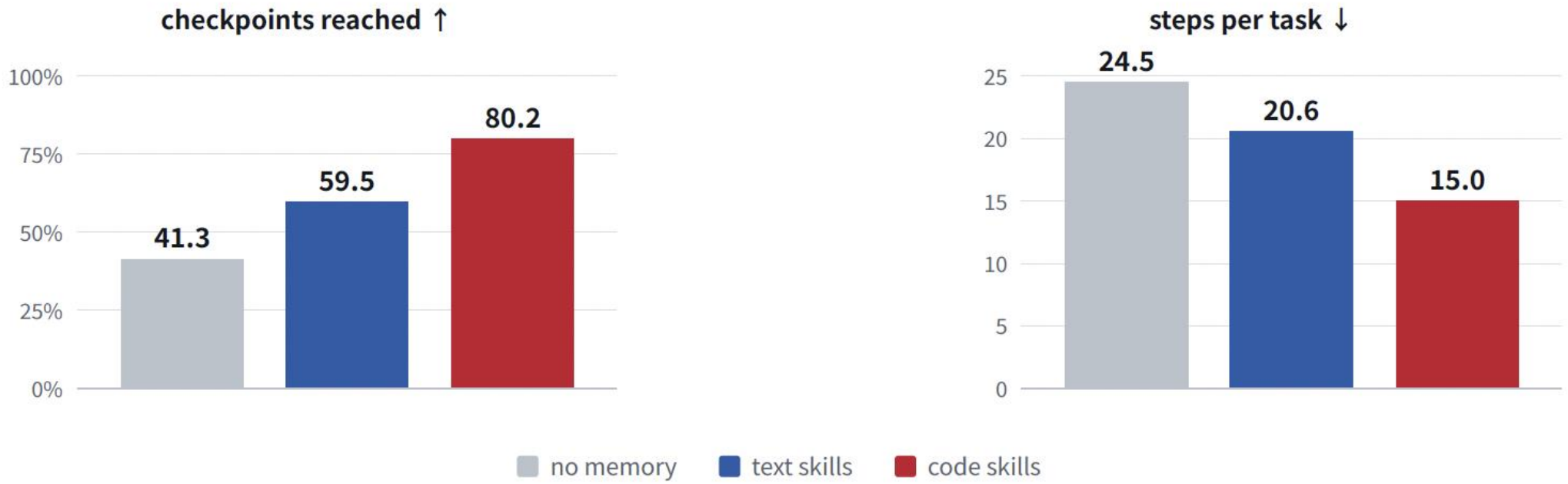
Task Compression with Code Skills

I recently moved. Can you change my billing address to "231 Willow Way, Suite 100, Chicago, IL, 60601"? Then, update my shipping address to: 987 Sycamore Circle, Philadelphia, PA, 19102.



[1] Inducing Programmatic Skills for Agentic Tasks. Wang et al., COLM 2025.

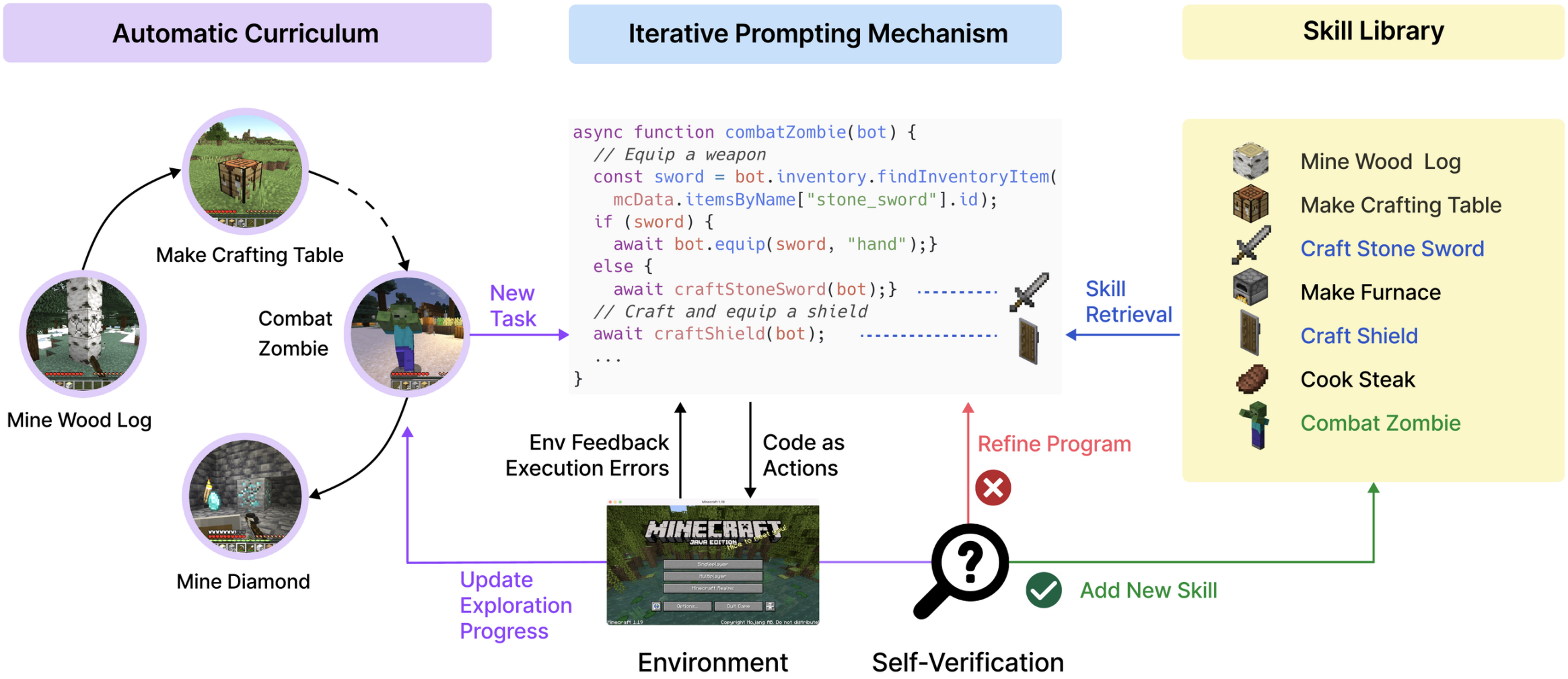
Code Skill vs Text Skill



- Comparison between text and code skills on web tasks with repeated structure
- Code skills generally improve success (checkpoints reached) over text skills
- They also use fewer steps by the agent

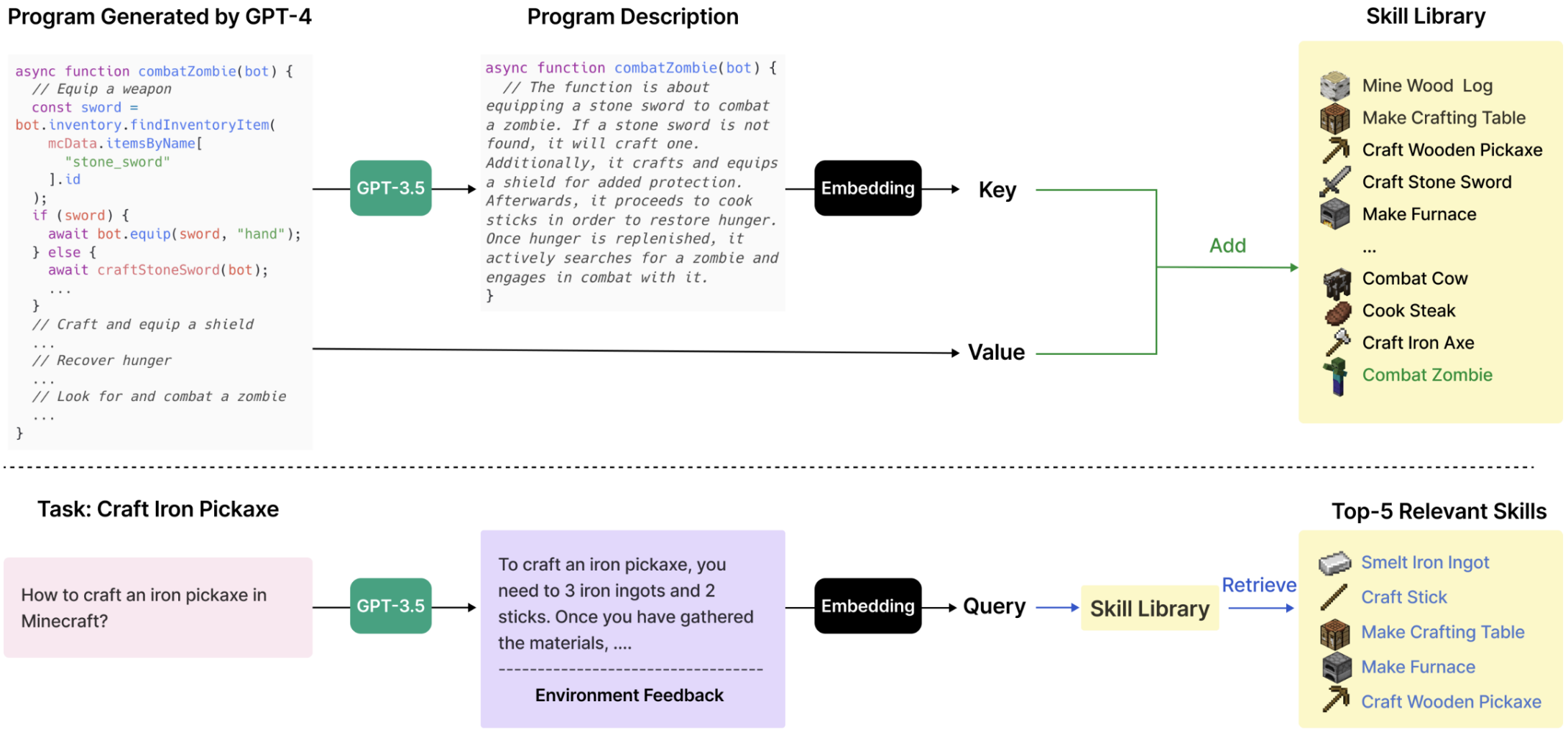
[1] Inducing Programmatic Skills for Agentic Tasks. Wang et al., COLM 2025.

Code Skills for Game Agent



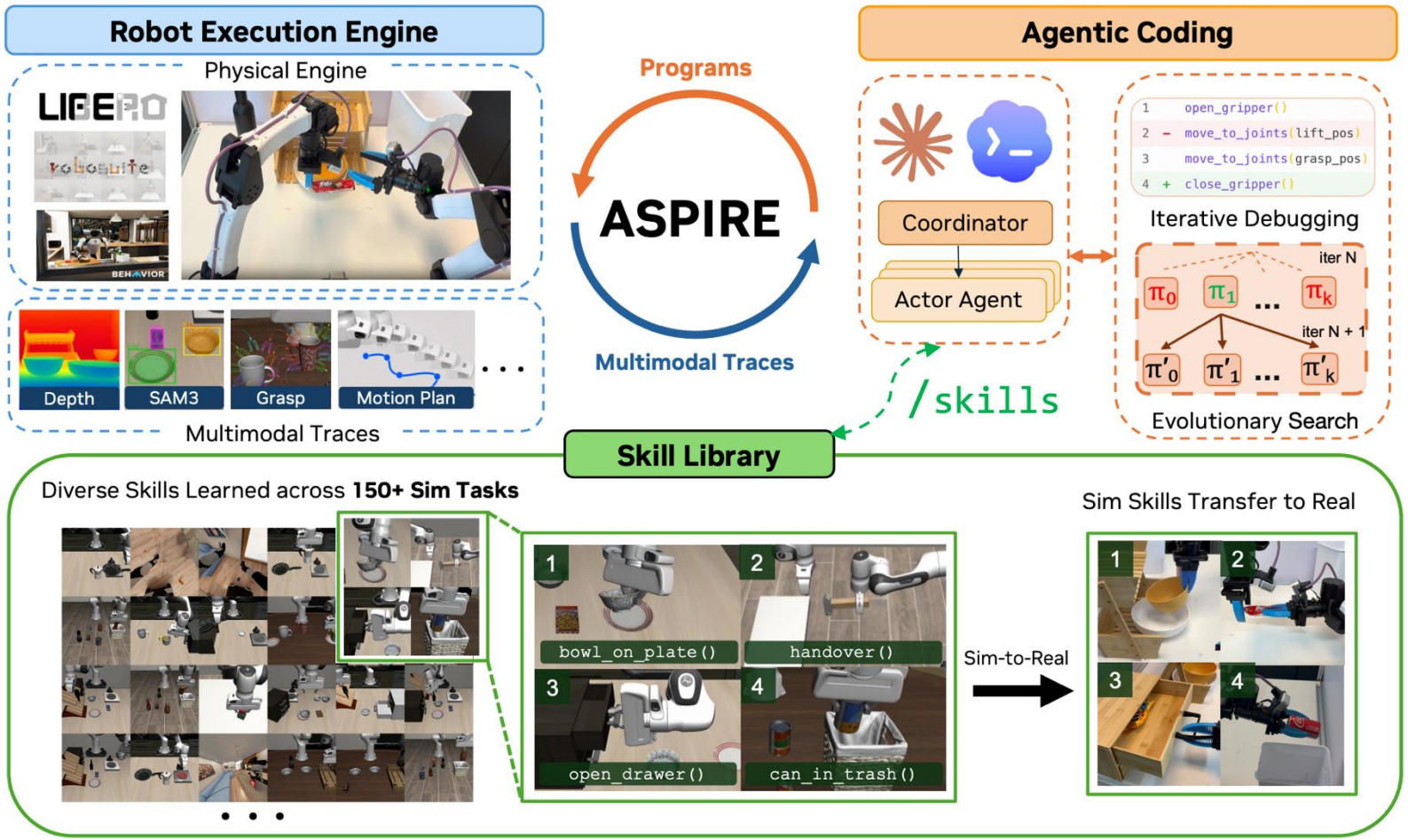
[1] Voyager: An Open-Ended Embodied Agent with Large Language Models. Wang et al., NeurIPS 2023.

Code Skills for Game Agent



[1] Voyager: An Open-Ended Embodied Agent with Large Language Models. Wang et al., NeurIPS 2023.

Code Skills for Embodied Agent



(c) ★ Multi-Angle Approach

fix_code.py

```python
for angle_deg in [180, -90, 90, -45, 45]:  # south, sides only
    angle = np.radians(angle_deg)
    tx = radio_pos[0] + 0.7 * np.cos(angle)
    ty = radio_pos[1] + 0.7 * np.sin(angle)
    face_yaw = np.arctan2(radio_pos[1]-ty, radio_pos[0]-tx)
    moved = safe_navigate([tx, ty, face_yaw], f"ang_{angle_deg}")
    if moved and dist_to(radio_pos[:2]) < 0.8:
        break
...
```

New Skill

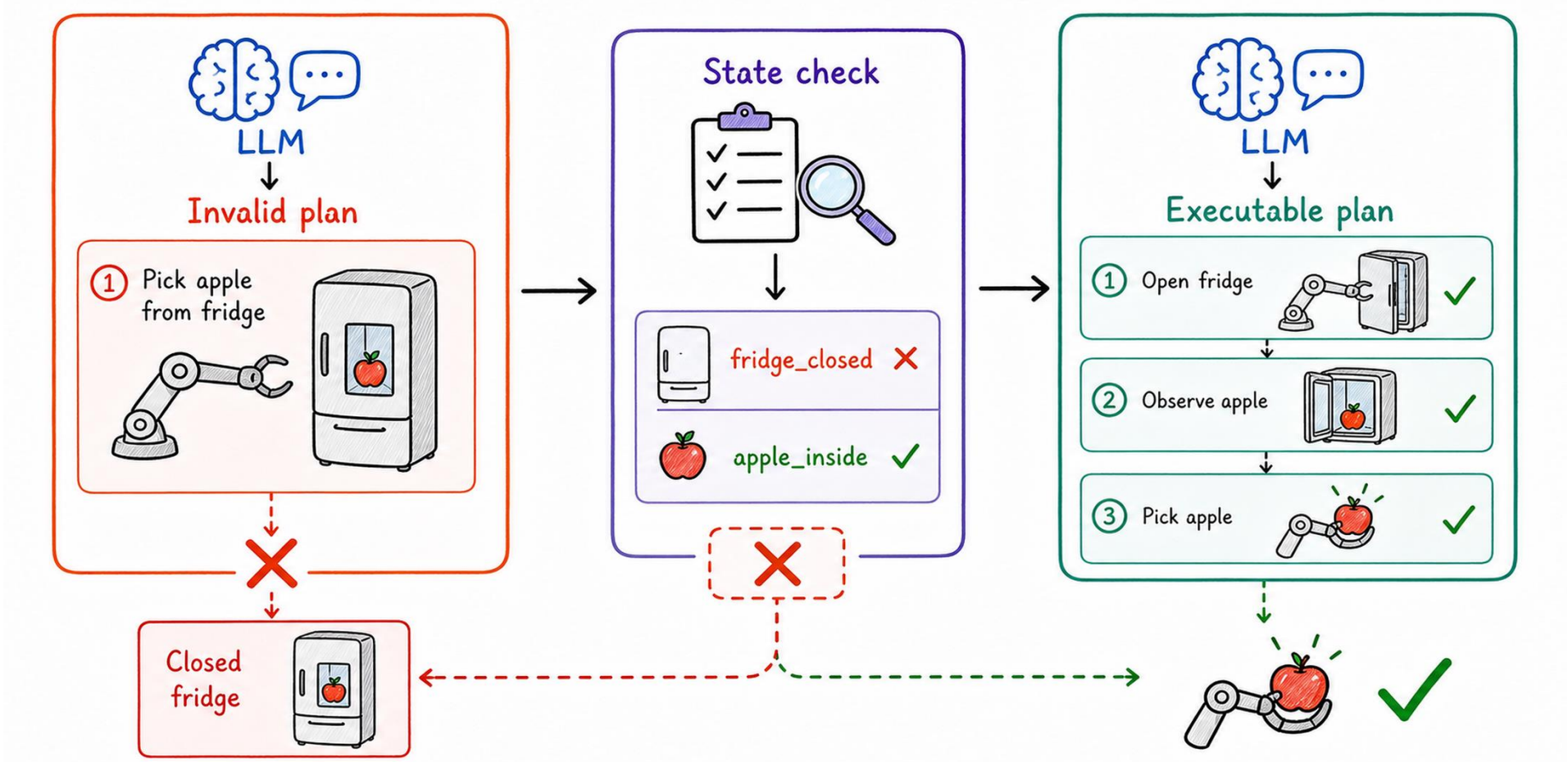
If cuRobo returns PLANNING_ERROR for nearby goals (inside the collision-avoidance buffer), try rotating the approach vector by 45°, 90°, or 180°.

Each angle puts the goal on a different side of the object.
When one side is blocked, another is often open.

[1] ASPIRE: Agentic /Skills Discovery for Robotics. Nvidia 2026.

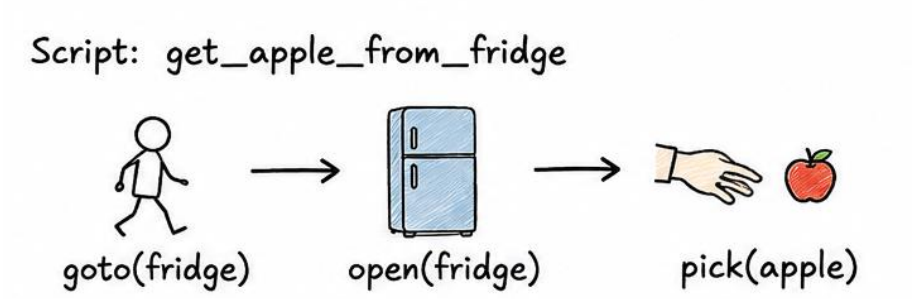
Neuro-Symbolic Skills for Embodied Agent

Embodied Planning：模型不仅需要生成合理的规划，还需要满足环境状态与物理约束，例如动作前提、可达性、affordance、安全约束等



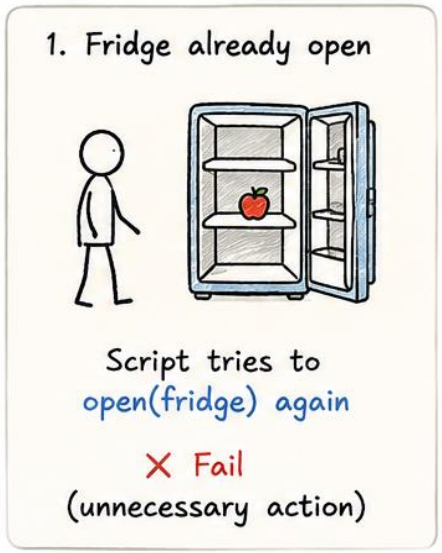
Skills for Embodied Agent

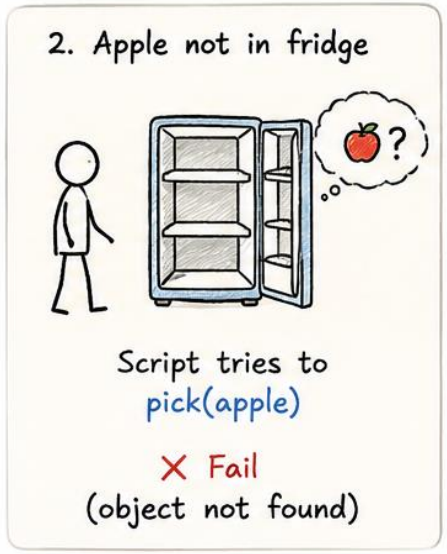
已有方法通常把交互轨迹压缩成**参数化动作脚本**

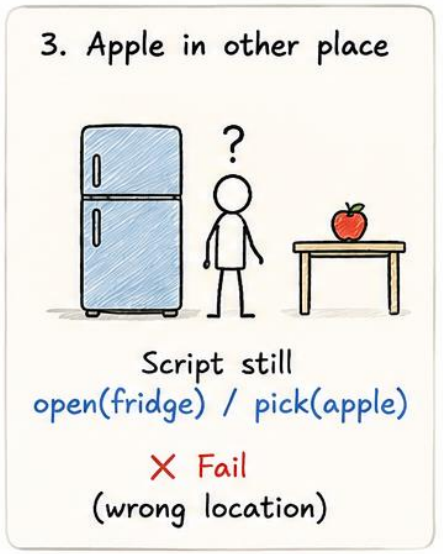


State-Blind Skill

- 缺少显式条件判断：不知道何时 open / skip
- 缺少动态变量绑定：目标对象依赖当前状态
- 环境偏移时容易失败，长程任务错误累积



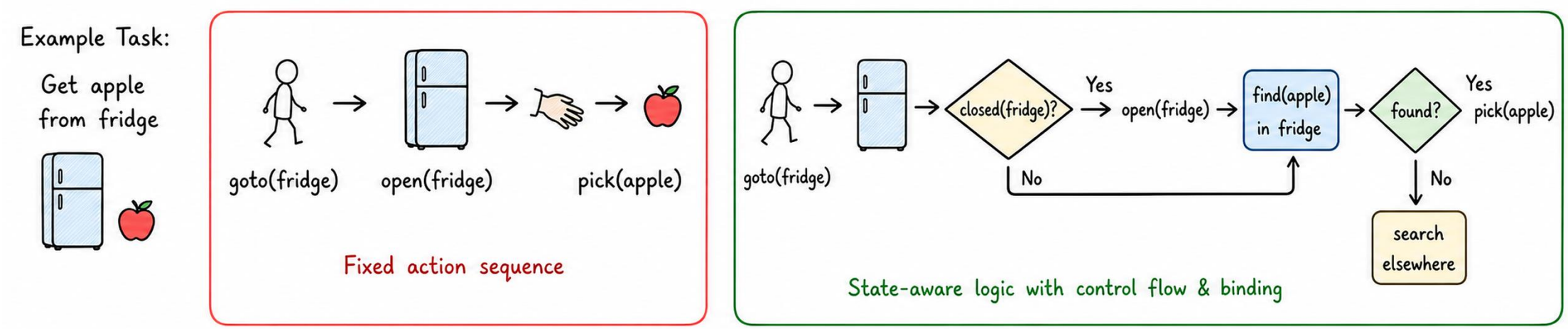




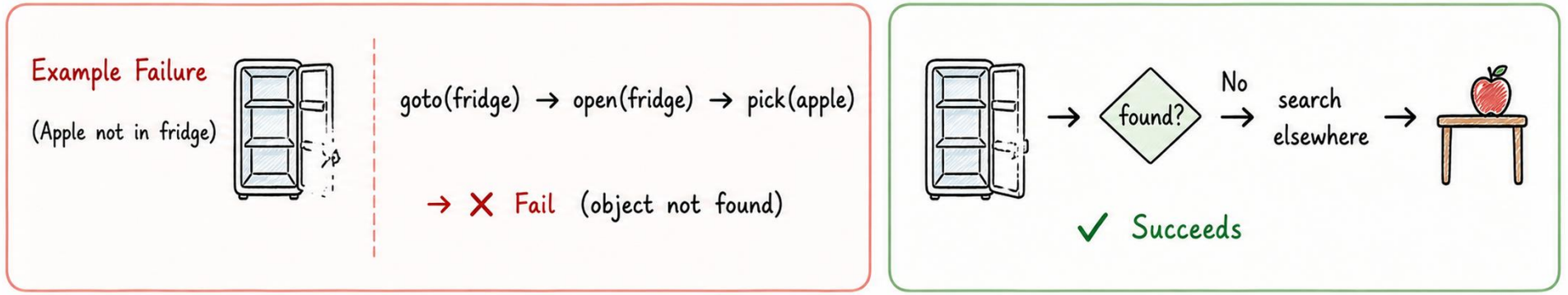


Neuro-Symbolic Skill

Neuro-Symbolic Skill： “状态谓词 + 控制流 + 动态变量绑定”的可执行图

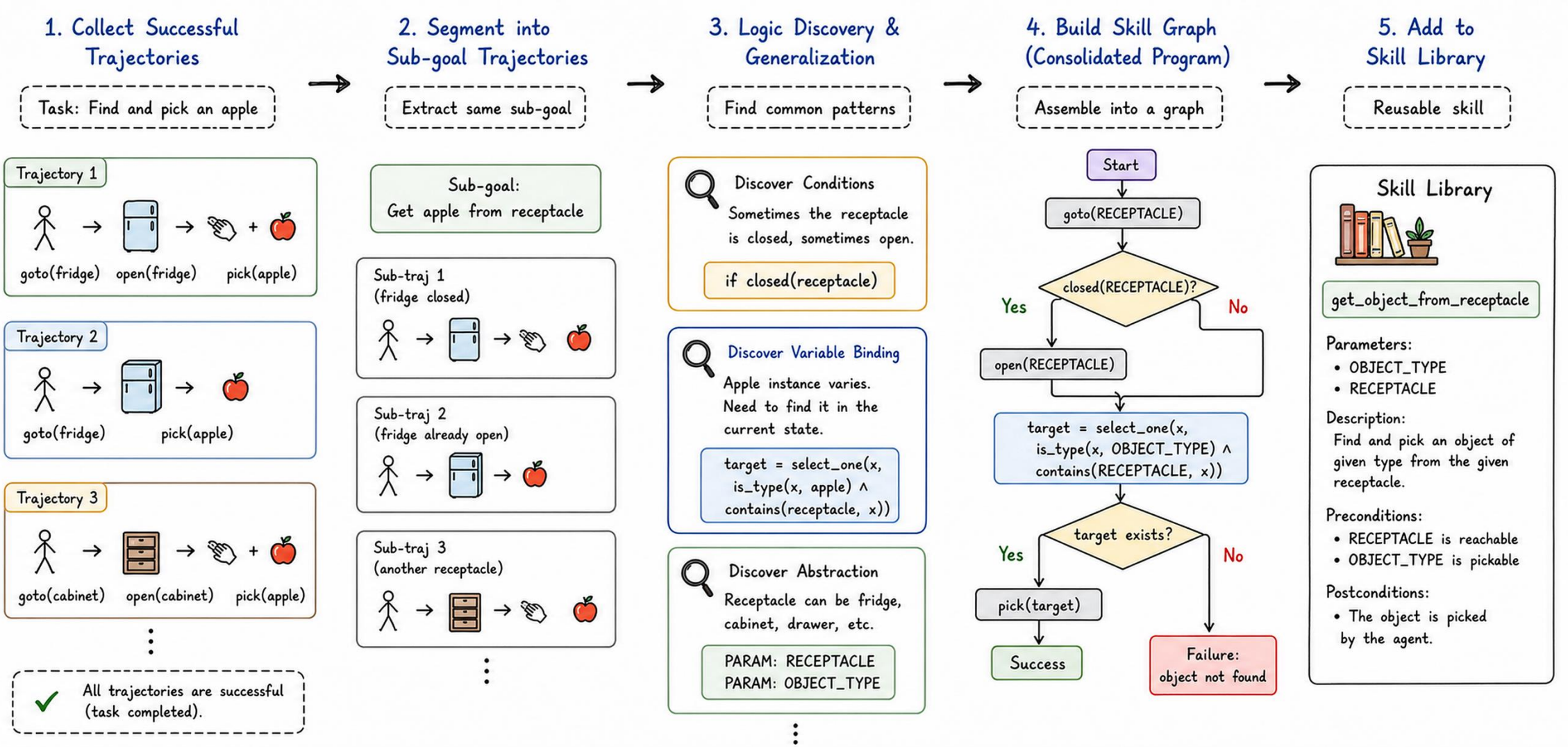


Neuro-Symbolic Skill Learns When, Why, and How to act



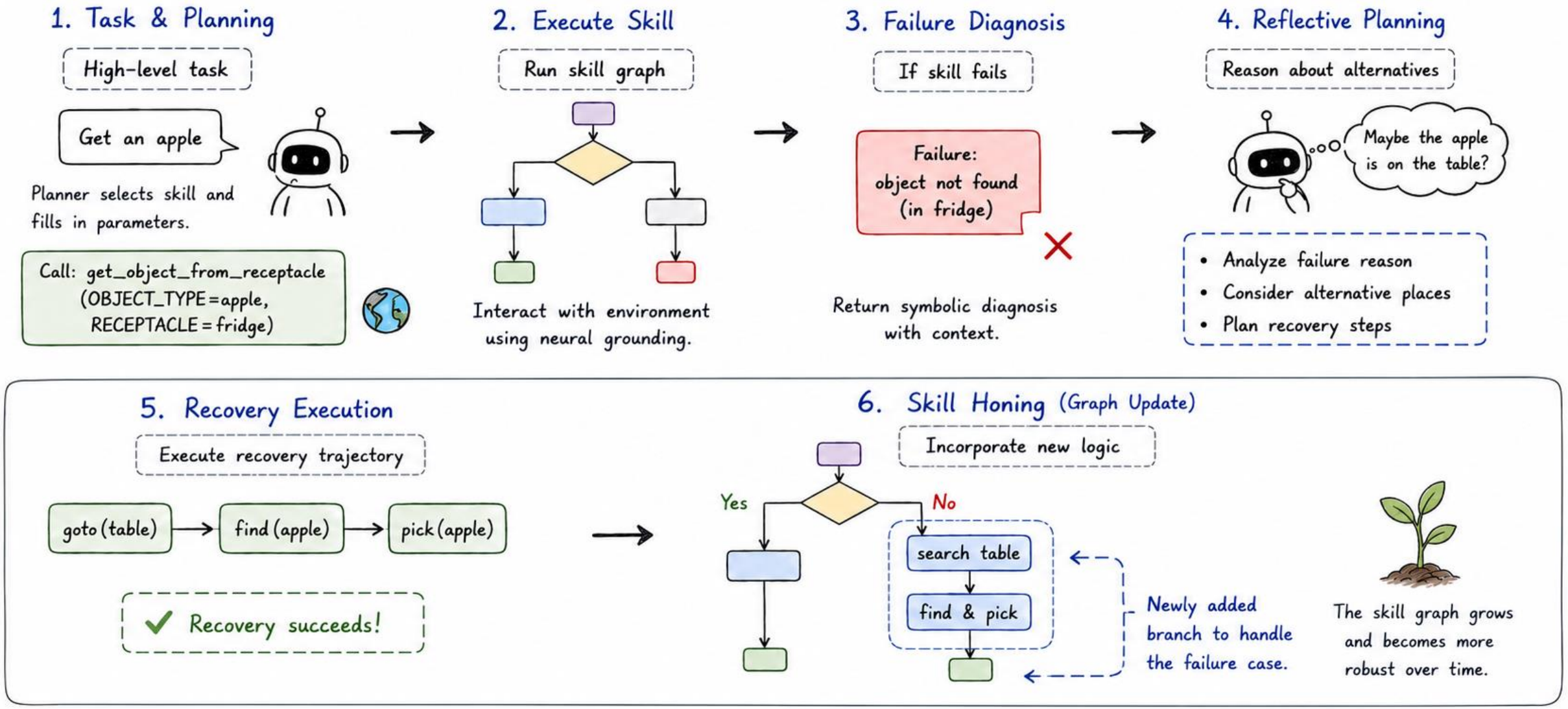
[1] Lifting Traces to Logic: Programmatic Skill Induction with Neuro-Symbolic Learning for Long-Horizon Agentic Tasks. Shao et al., ICML 2026.

Offline Skill Induction



[1] Lifting Traces to Logic: Programmatic Skill Induction with Neuro-Symbolic Learning for Long-Horizon Agentic Tasks. Shao et al., ICML 2026.

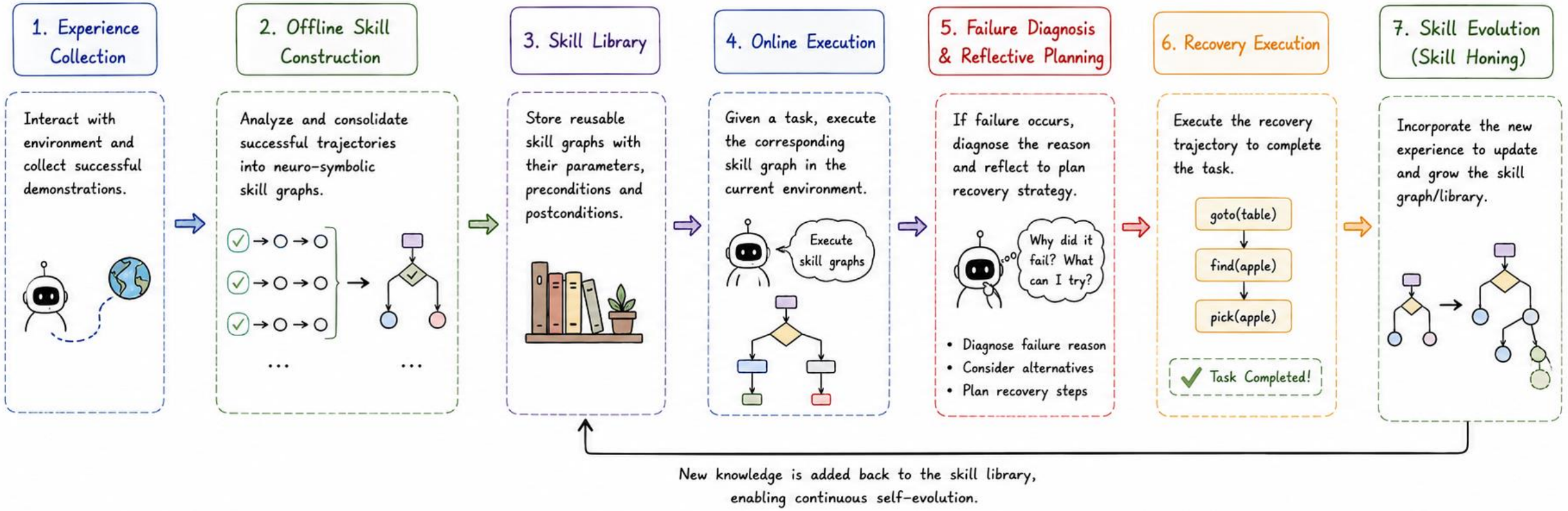
Online Skill Evolution



[1] Lifting Traces to Logic: Programmatic Skill Induction with Neuro-Symbolic Learning for Long-Horizon Agentic Tasks. Shao et al., ICML 2026.

Neuro-Symbolic Skill

将智能体的成功交互轨迹从状态盲的动作脚本，提升为包含神经感知Grounding、符号变量绑定、显式控制流和失败修复机制的逻辑驱动技能图



[1] Lifting Traces to Logic: Programmatic Skill Induction with Neuro-Symbolic Learning for Long-Horizon Agentic Tasks. Shao et al., ICML 2026.

Experimental Results

相比Text Skill、Code Skill能获得稳定的性能提升

Method	ALFWorld Success Rate (%)	WebShop Score(%)	WebShop Success Rate (%)	TextCraft Success Rate (%)
ReAct (Yao et al., 2022b)	85.8	44.0	20.0	62.0
Reflexion (Shinn et al., 2023)	84.3	40.8	23.0	59.0
ADaPT (Prasad et al., 2024)	67.9	45.8	29.0	72.5
StateAct (Rozanov & Rei, 2025)	84.3	40.6	8.00	83.0
$AWM_{offline}$ (Wang et al., 2025c)	88.6±1.1	51.8±1.7	32.2±1.3	66.3±2.0
AWM (Wang et al., 2025c)	91.3±0.8	49.2±1.9	30.0±2.0	92.5±3.6
ASI (Wang et al., 2025b)	70.6±1.9	7.7±1.7	7.50±3.0	77.8±1.8
NSI w.o. online honing	93.5±1.9	58.8±1.8	30.5±1.5	78.5±2.5
NSI (Ours)	**98.0±1.2**	**76.5±1.2**	**44.5±1.5**	**95.2±0.8**

[1] Lifting Traces to Logic: Programmatic Skill Induction with Neuro-Symbolic Learning for Long-Horizon Agentic Tasks. Shao et al., ICML 2026.

Agentic Loop:

Planning & Execution Control

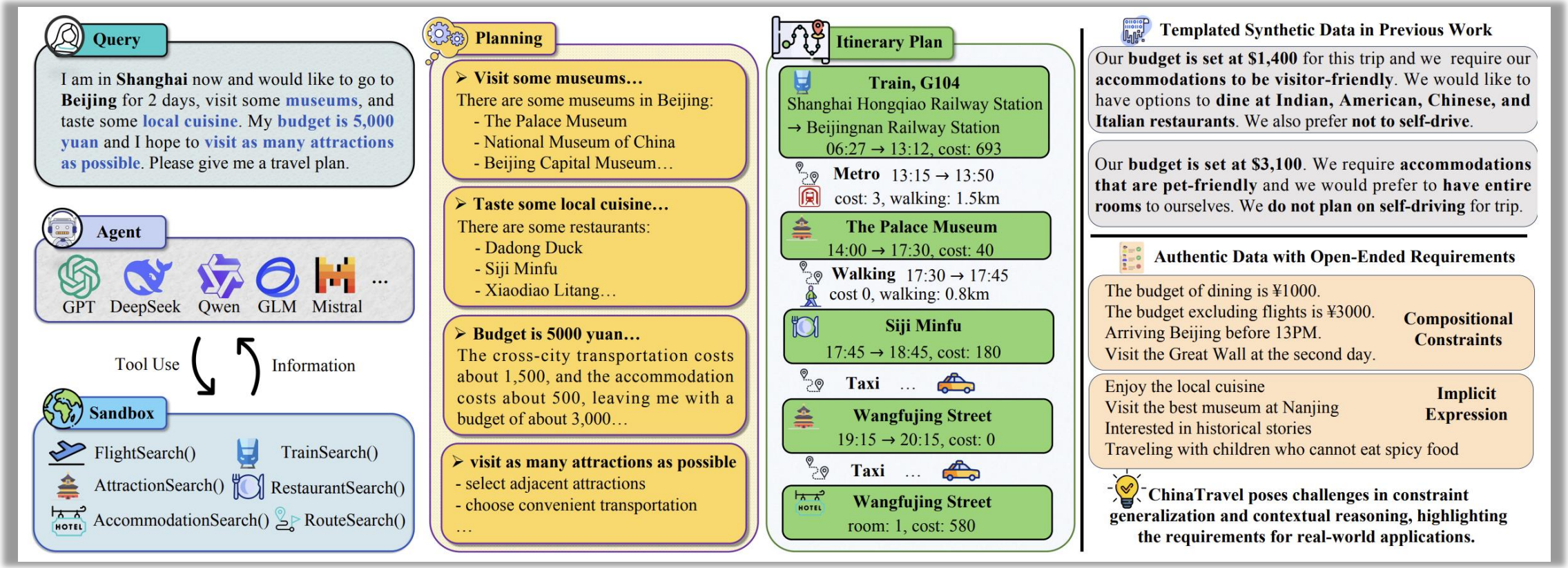
Agentic Loop



Planning in Agentic Tasks

Task

Given a user's travel requirements, the task is to generate a valid travel plan that satisfies all constraints



Chain-of-Thought is a plan

A CoT reasoning step

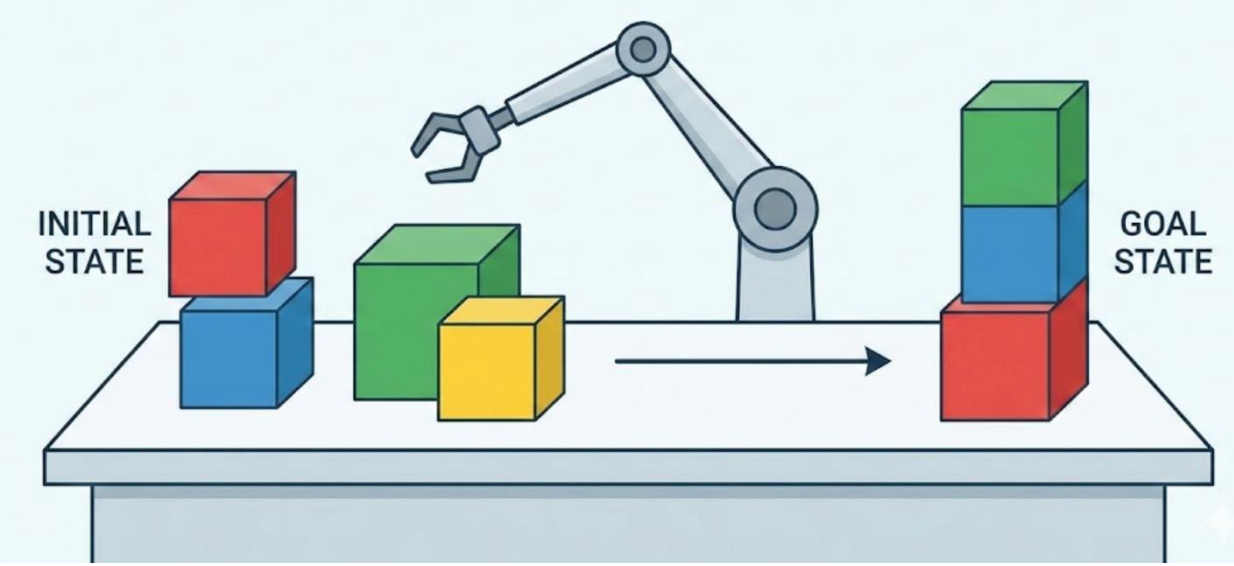
- changes only text
- can be undone by writing more
- every intermediate is visible
- plans are useful if the model can condition on them better

An action in the world

- changes the state the agent meets next
- can reveal information
- can fail
- can be irreversible: submit, send

Acting means plans must handle feasibility, consequences, information, and recovery

Chain-of-Thought is a plan



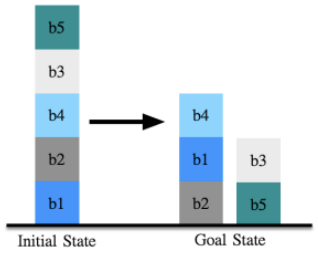
Plan **a sequence of robotic arm actions**

to move the block from the **initial state**

to the **target state**

A Failure Example of GPT-4 in Planning

Problem (P1): You have 5 blocks. One cannot place more than one block on another block. b5 is on top of b3. b4 is on top of b2. b2 is on top of b1. b3 is on top of b4. b1 is on the table. b5 is clear. Your arm is empty.
Your goal is to move the blocks.
b1 should be on top of b2.
b3 should be on top of b5.
b4 should be on top of b1.

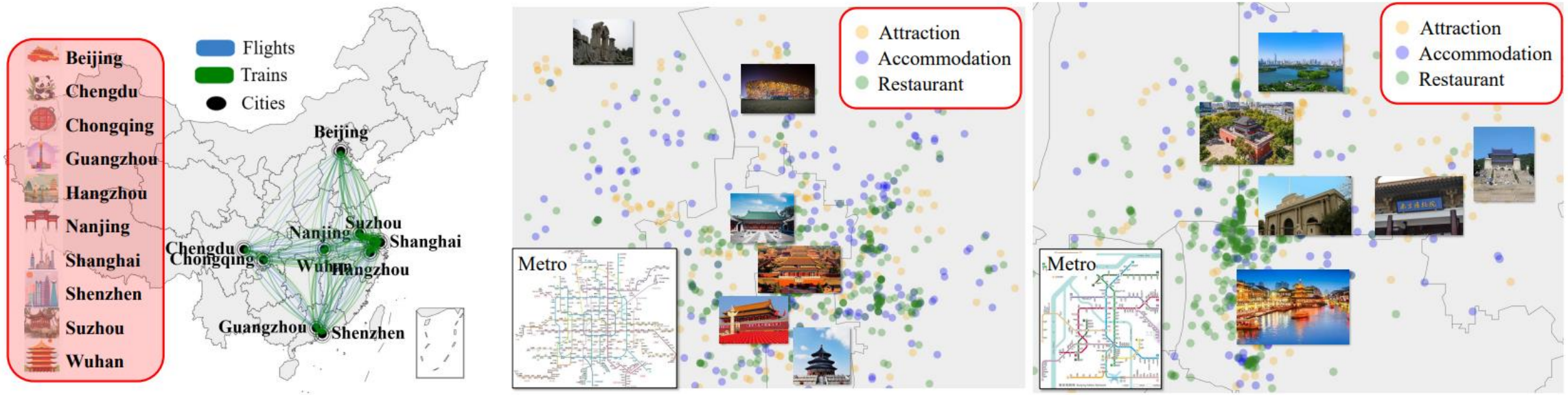


GPT-4:

```
1. Move b5 from b3 to the table.
2. Move b3 from b4 to b5.
3. Move b4 from b2 to the table.
4. Move b2 from b1 to b5.
5. Move b1 from the table to b2.
6. Move b4 from the table to b1.
```

ChinaTravel: A Travel Planning Benchmark

- **GDP前10城市**：上海、北京、深圳、重庆、广州、苏州、成都、杭州、武汉、南京
- **6.5K+ 跨城交通**：720架次飞机 、5770班次火车……
- **3K+ 景点**：人文景观、自然景观、美术馆、博物馆、纪念馆、公园、游乐园……
- **4K+ 宾馆**：洗衣服务、充电桩、停车场、宠物友好、会议厅、泳池、健身房……



[1] ChinaTravel: An Open-Ended Travel Planning Benchmark with Compositional Constraint Validation for Language Agents. Shao et al. ICLR 2026.

Travel Requirements

- **需求收集**：团队收集来自**300+人的真实自然语言需求**，构建了以16个符号概念为基础的约束需求
- **指标定义**：包含 **游、食、住、行**四个基础视角，构建了**23条可行性指标**，基本覆盖日常出游需要

信息类型	具体信息
基础信息	游玩天数、游玩人数、总开销
游	景点类型、景点名称
食	食物类型、餐馆名称、 人均单次餐饮价格
住	酒店特征、酒店名称、 房间个数、房间类型、酒店房间价格
行	市内交通方式、 城际交通方式、火车类型

可行性指标	
基础信息	游玩天数、游玩人数、总开销
游	选择景点在目标城市存在、在景点的开放时间进行访问、在旅程中访问景点不重复、景点的门票信息正确……
食	选择餐馆在目标城市存在、在餐馆的开放时间进行访问、在旅程中访问餐馆不重复、餐馆的人均价格信息正确……
住	选择酒店在目标城市存在、房间信息满足人数需求……
行	选择交通方式是存在的、给出交通方案的价格正确、给出交通方案的时间正确、给出交通方案的距离正确……
时间	给出的规划方案中每个事件按时间顺序发生，无冲突
空间	在不同地点的事件安排都需要保证给出可行的交通方式

[1] ChinaTravel: An Open-Ended Travel Planning Benchmark with Compositional Constraint Validation for Language Agents. Shao et al. ICLR 2026.

ChinaTravel: Domain-Specific Language (DSL)

LLM友好、可组合、可验证

- **类 Python 语法**：可组合的原子概念（空间、时间、成本、类型……）
- **程序化验证**：可行性 ∧ 逻辑约束 ∧ 优先级排序
- **可组合泛化**：新约束可由少量基本概念组合而成
- **可扩展性强**：数百种人类需求约束，可通过约 20 个原子概念表达

判断餐饮预算是否超过1000

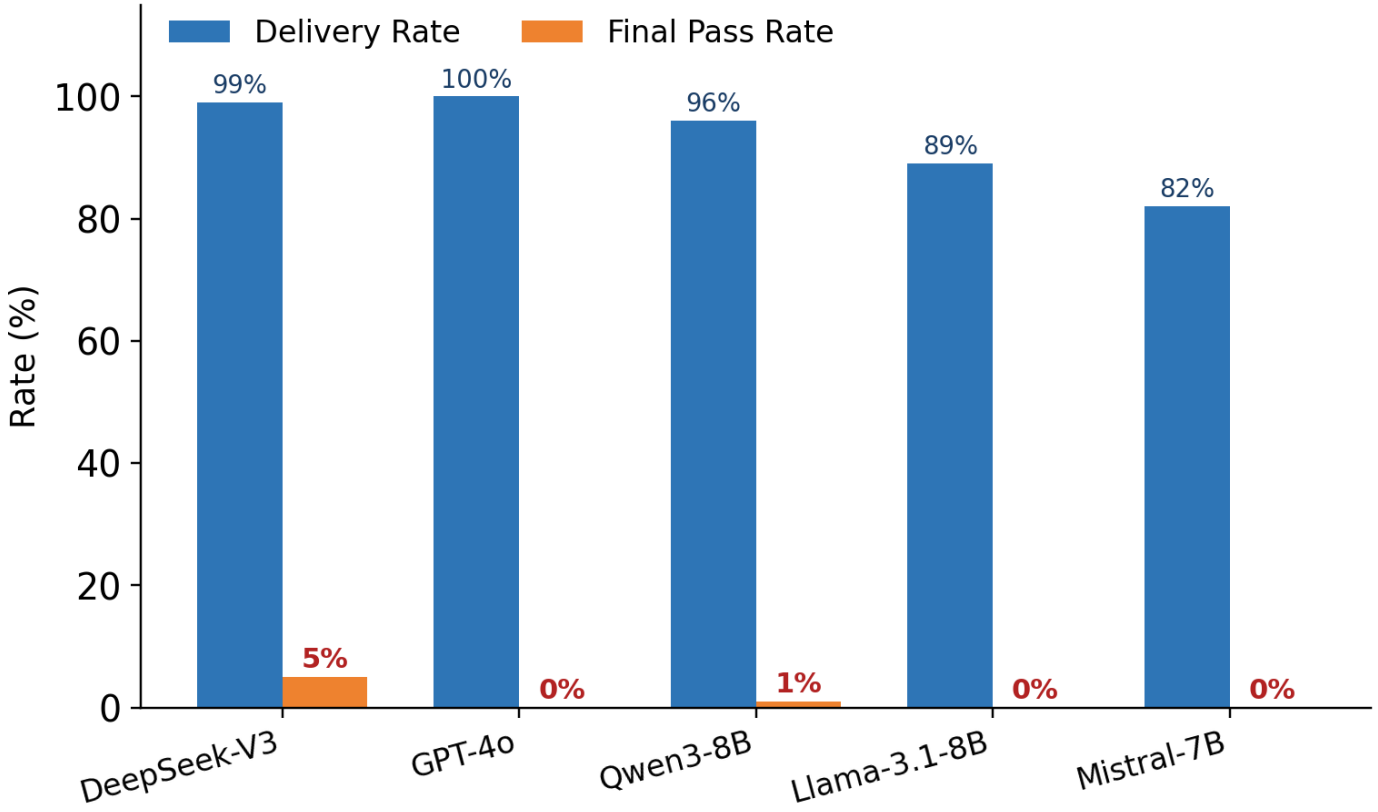
```
# Dining expenses <= 1000 CNY.
dining_cost = 0
for act_i in allactivities(plan):
  typ = activity_type(act_i)
  if typ=="breakfast" or typ=="lunch" or
       typ=="dinner": dining_cost =
      dining_cost + activity_cost(act_i)
return dining_cost <= 1000
```

判断参观的景点总数量

```
# Arriving in Shanghai should be before
    6 PM on the second day.
return_time = 0
for act_i in day_activities(plan, 2):
  typ = activity_type(act_i)
  dest = transport_destination(act_i)
  if (typ=="train" or typ=="airplane")
      and des=="Shanghai": return_time
      == activity_endtime(act_i)
return return_time < "18:00"
```

[1] ChinaTravel: An Open-Ended Travel Planning Benchmark with Compositional Constraint Validation for Language Agents. Shao et al. ICLR 2026.

Results of CoT Planning



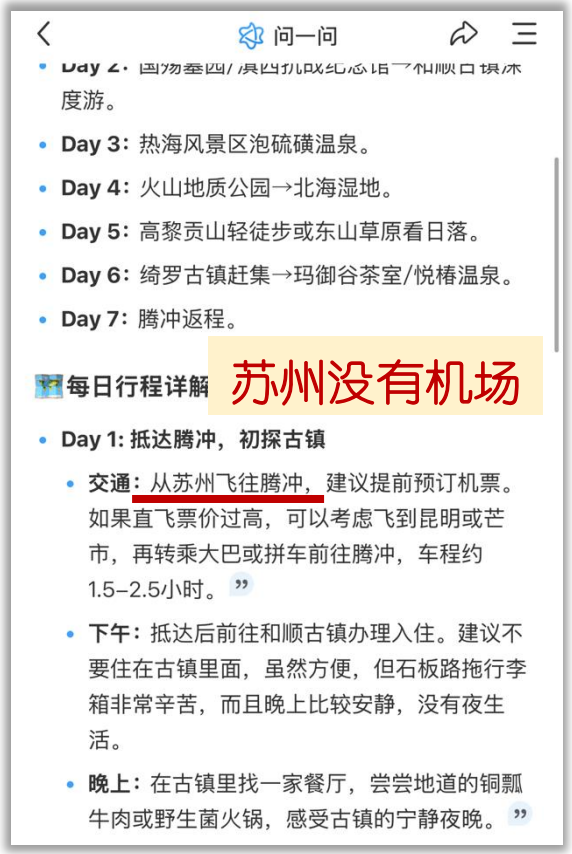
- Structured output ≠ valid plan: **High DR masks low FPR**
- GPT-4o: **0% FPR**
- DeepSeek-V3: 5% on Easy, **~0% on human splits**

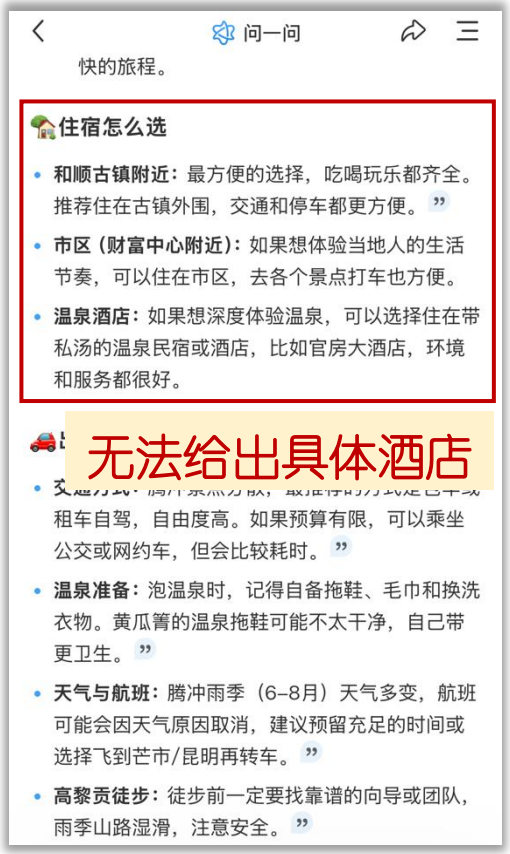
[1] ChinaTravel: An Open-Ended Travel Planning Benchmark with Compositional Constraint Validation for Language Agents. Shao et al. ICLR 2026.

Some Failure Cases

用户：帮我规划一个苏州到云南腾冲、芒市5日游的规划

信息错误、仅能粗粒度规划



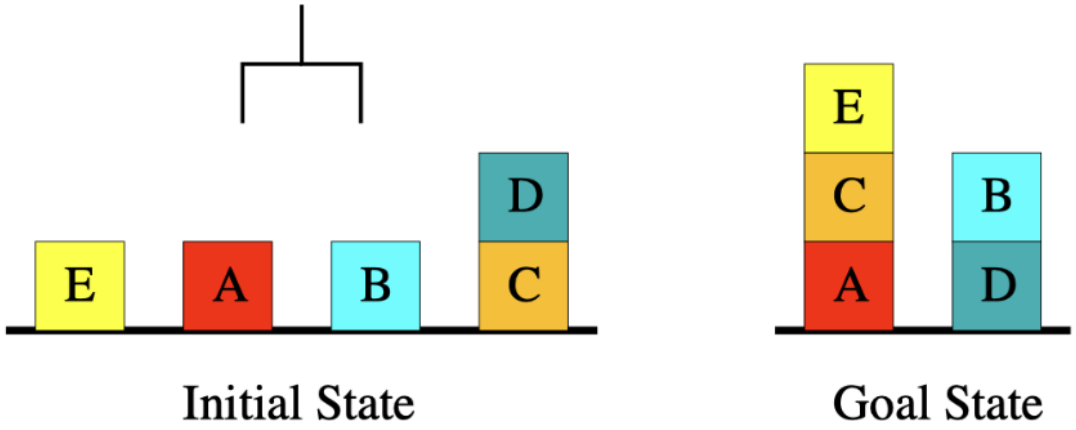


规划方案不可行、不合理





Classical Symbolic Planning



- A state is a set of **predicates**; actions have **preconditions** and **effects**
- A valid sequence of actions satisfies every action‘s preconditions
- **Planning is a search problem**: find a sequence of valid actions that reach the goal

state

```
on-table(x) · on(x, y)
clear(x) · holding(x)
hand-empty
```

action

```
pick-up(x) · unstack(x, y)
put-down(x) · stack(x, y)
```

preconditions + effects

```
pick-up(x): on-table(x) ∧ clear(x) ∧ hand-empty
→ holding(x) ∧ ¬on-table(x) ∧ ¬hand-empty
```

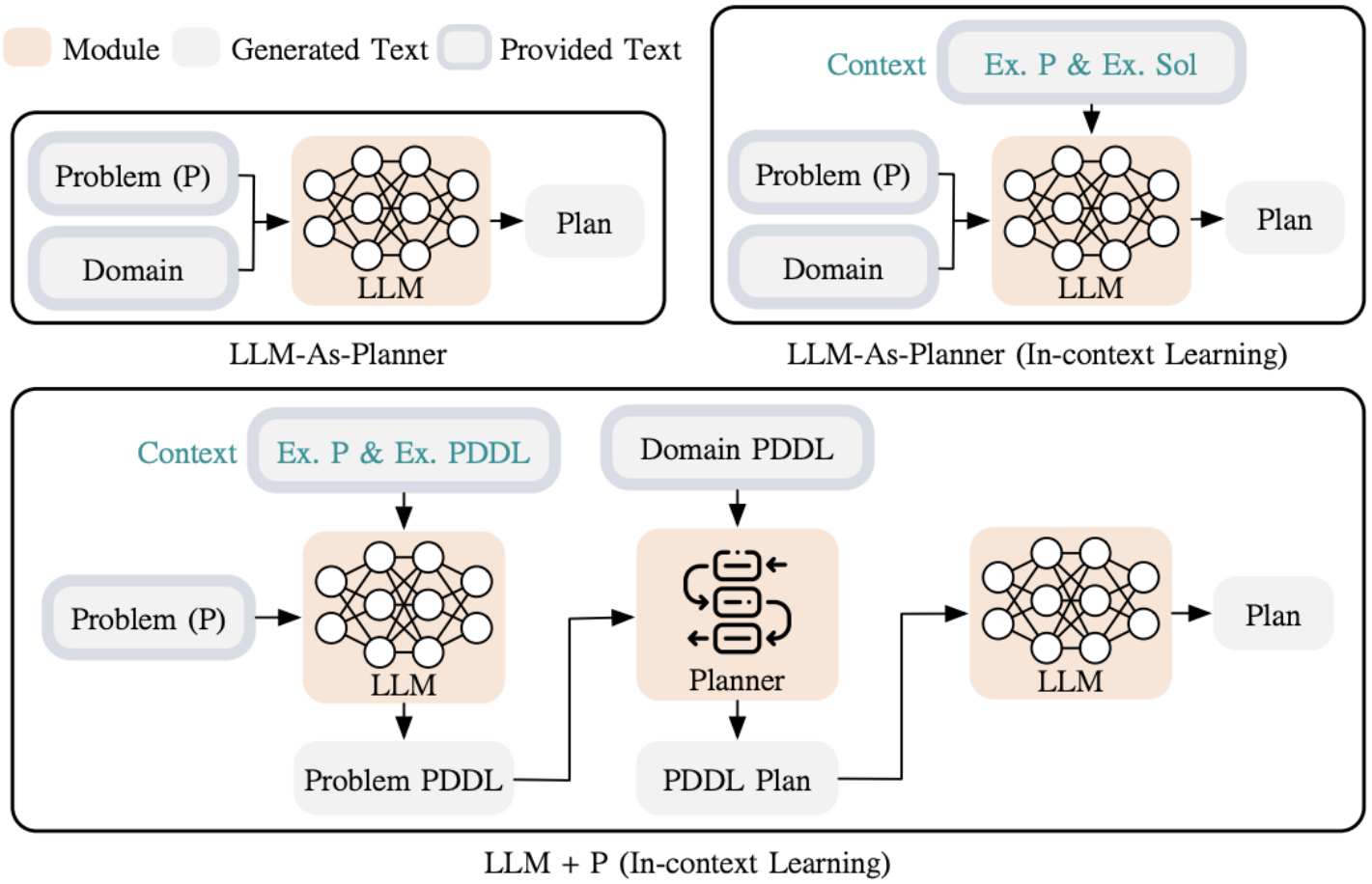
initial state + goal

```
on(blue, orange) · clear(red, blue, yellow)
goal: on(orange, blue)
```

plan

```
unstack(blue, orange)
put-down(blue)
pick-up(orange)
stack(orange, blue)
```

LLM + Symbolic Planner



Tidy-Up Problem PDDL Generated by LLM+P

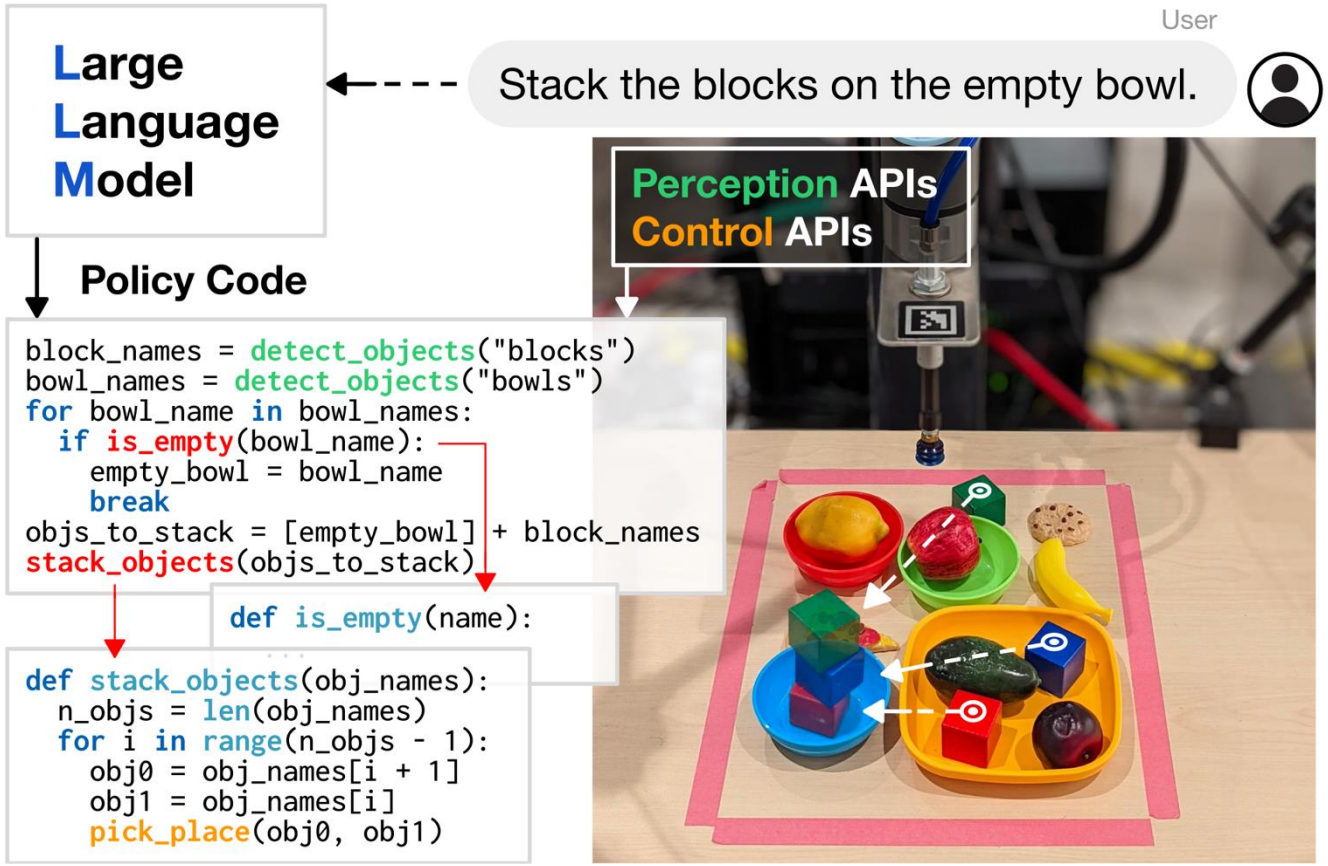
Problem (P): You are a home robot with one gripper. The distance between coffee table and side table is 10. The distance between coffee table and pantry is 20... You are at the coffee table. There is a mustard bottle... Your goal is to move objects to their destinations...

Problem PDDL generated by LLM+P:

```
(:objects coffee-table side-table
recycle-bin pantry - location
mustard-bottle soup-can - object)
(:init (= (total-cost) 0) (=
(distance coffee-table side-table)
10) (= (distance coffee-table
pantry) 20) ... (robot-at
coffee-table) (at mustard-bottle
coffee-table) (at soup-can
side-table) (hand-empty) )
(:goal (and (at mustard-bottle
pantry) (at soup-can recycle-bin)))
(:metric minimize (total-cost)) )
```

[1] LLM+P: Empowering Large Language Models with Optimal Planning Proficiency. Liu et al. Arxiv 2023.

Code-based Planning

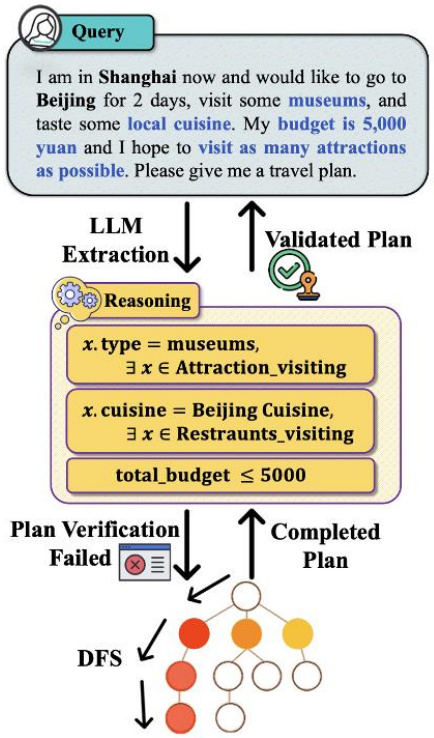


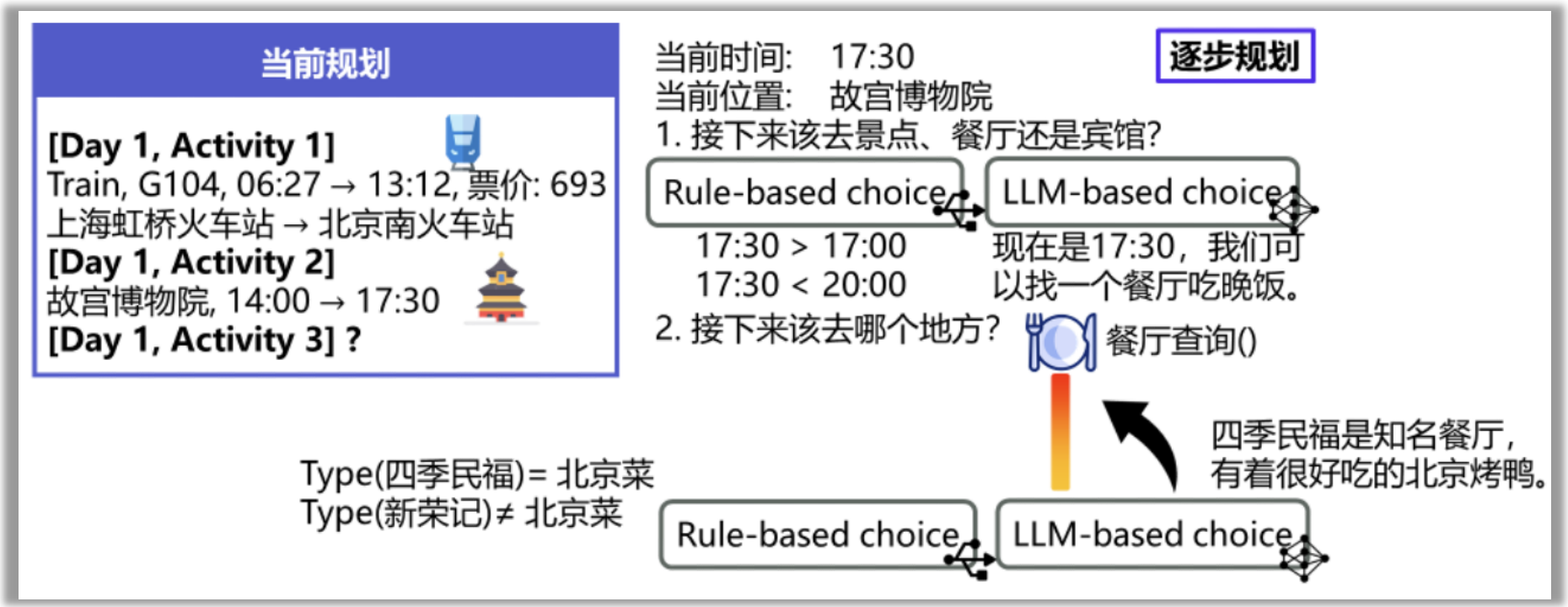
- **Formal plans are checkable; language plans are general. Programs are in-between**.
- Line of work in ~2023: use a code LLM to write code that calls perception and control APIs
- The program runs and can be inspected, but is limited by what its APIs and programmatic structure allow

[1] Code as Policies: Language Model Programs for Embodied Control. J. Liang et al. ICRA 2023.

Search Augmented Planning

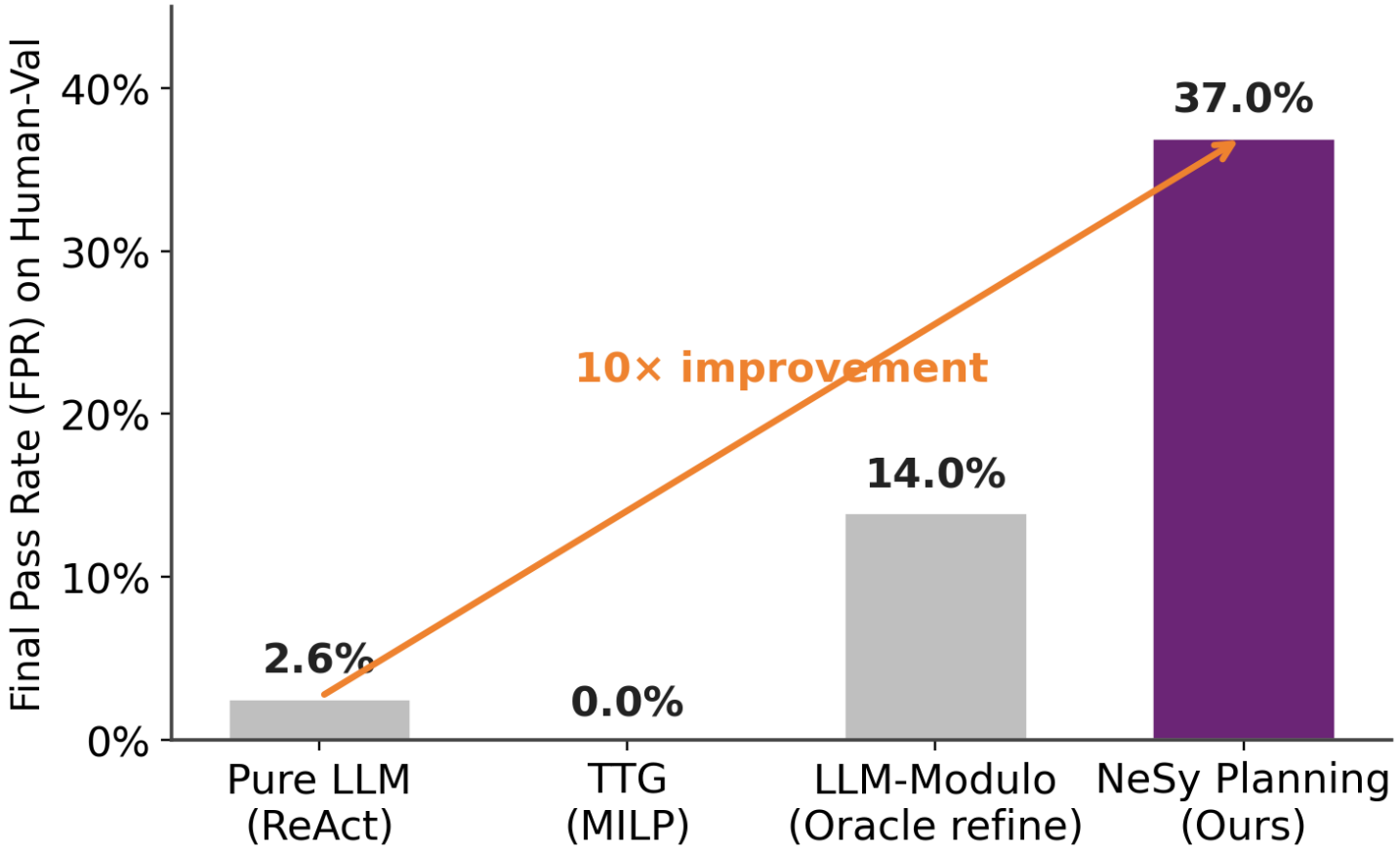
Natural Language -> DSL → LLM Guided Symbolic Search → DSL Verify LLM Revise





[1] ChinaTravel: An Open-Ended Travel Planning Benchmark with Compositional Constraint Validation for Language Agents. Shao et al. ICLR 2026.

Search Augmented Planning

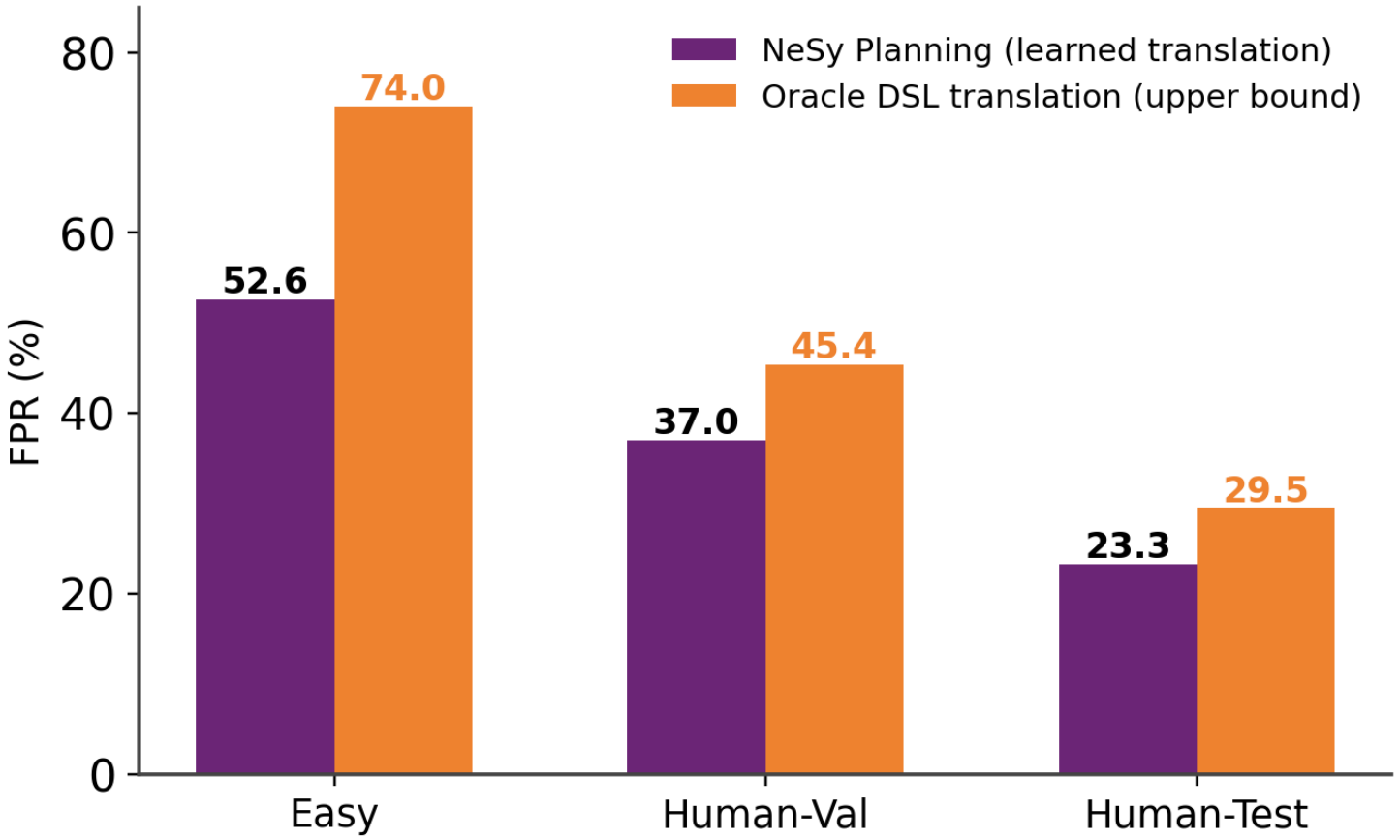


2.6% → 37.0%

Pure LLM → NeSy Planning

[1] ChinaTravel: An Open-Ended Travel Planning Benchmark with Compositional Constraint Validation for Language Agents. Shao et al. ICLR 2026.

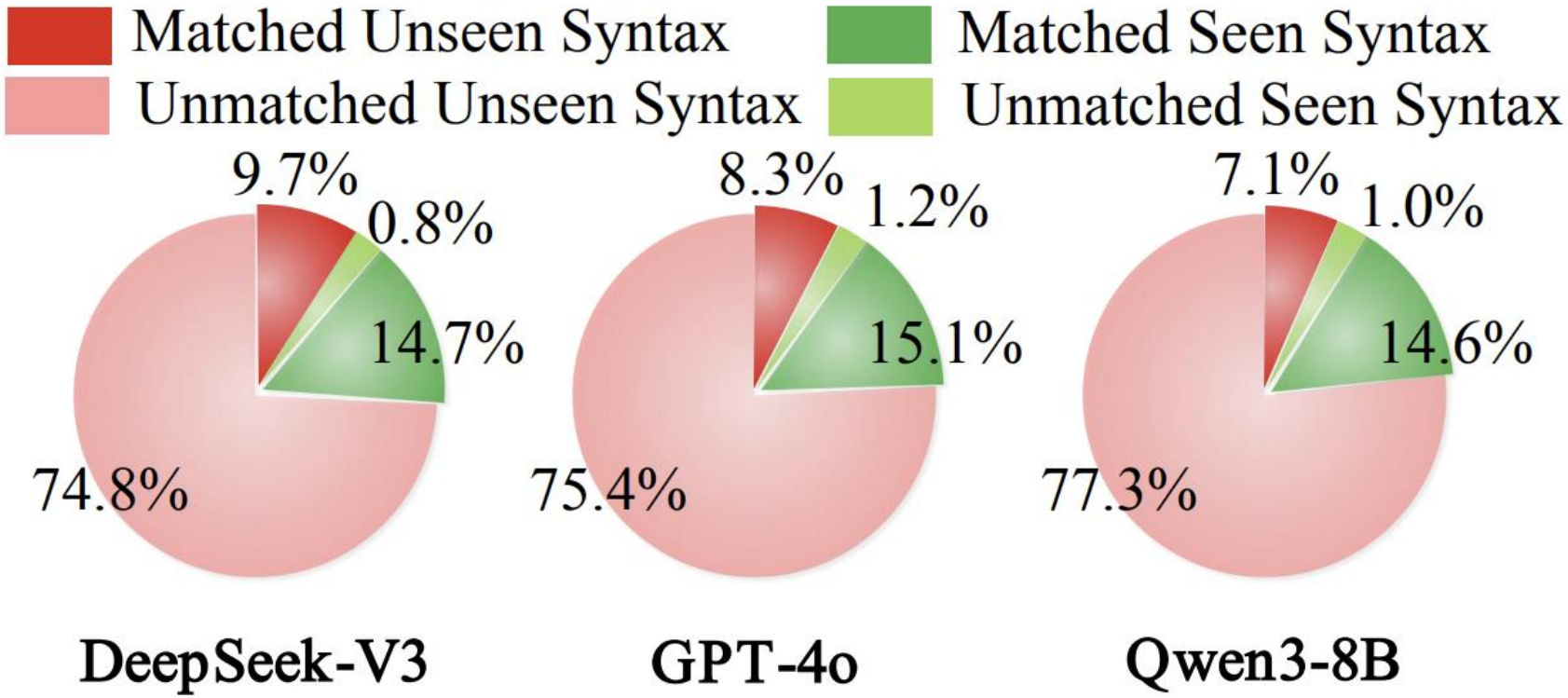
Challenge



- 给出正确的DSL语言，成功率可显著提升
- LLM在理解开放用户需求并将其翻译成符号语言上仍存在挑战
- 即便给定标准的DSL语言，符号求解器约束通过率也并非100%，搜索效率仍然是挑战

[1] ChinaTravel: An Open-Ended Travel Planning Benchmark with Compositional Constraint Validation for Language Agents. Shao et al. ICLR 2026.

Compositional Generalization



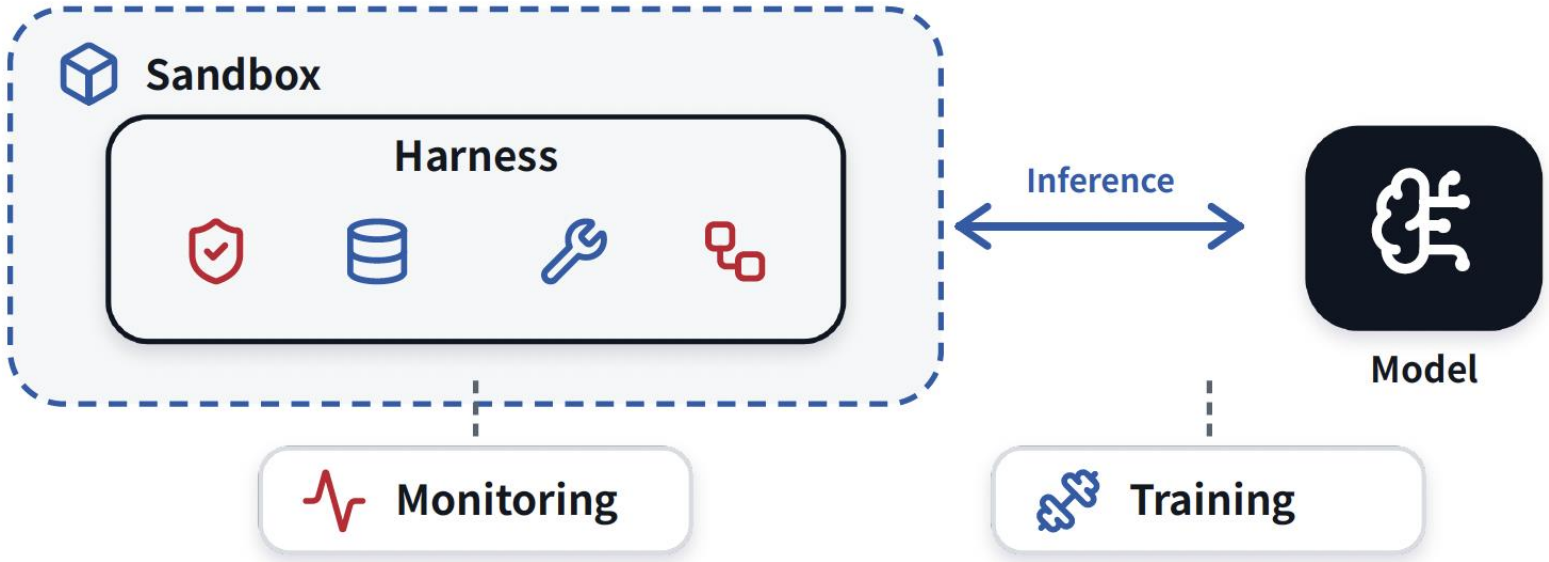
在开放式旅行规划中，用户会把时间、预算、类型、偏好等基础概念组合为训练阶段未见过的新约束，**LLM**难以实现稳定的组合泛化

[1] ChinaTravel: An Open-Ended Travel Planning Benchmark with Compositional Constraint Validation for Language Agents. Shao et al. ICLR 2026.

Summary

Agents are systems, not just models

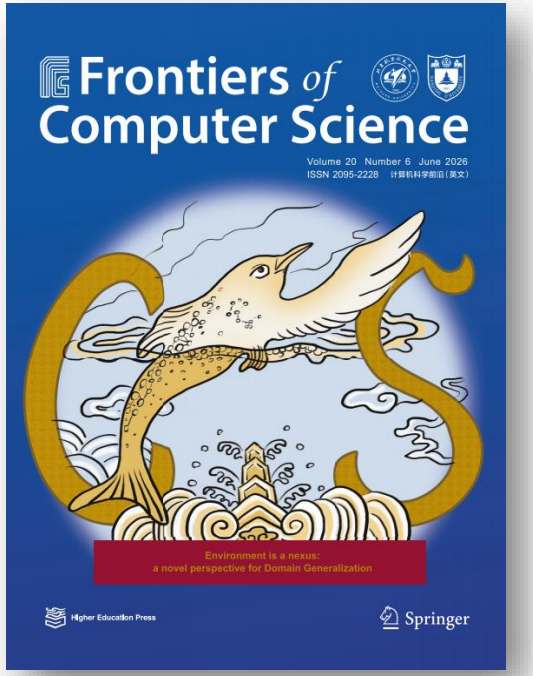
Symbolic methods power the agent harness



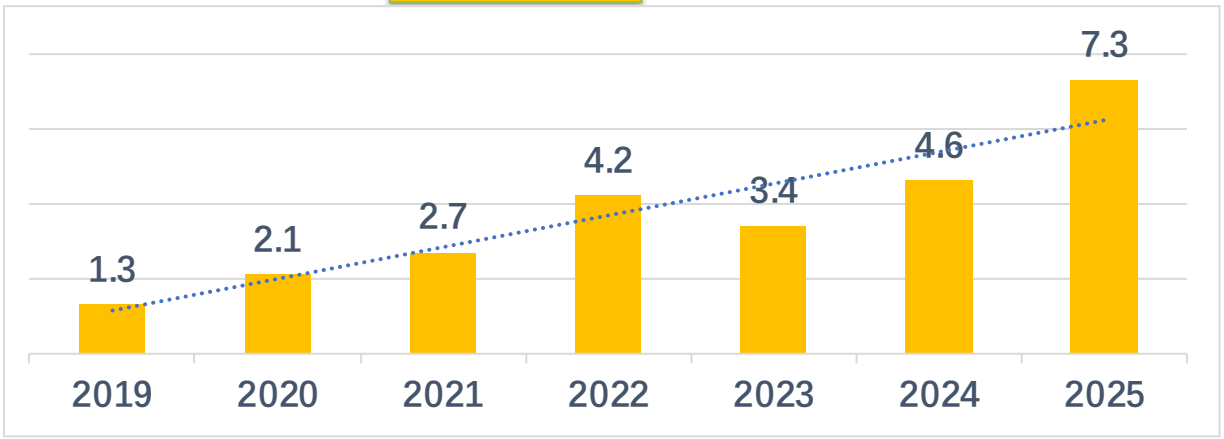
Frontiers of Computer Science 诚邀您投稿！

FCS Frontiers of Computer Science

➢ 2007年创刊，HEP与Springer联合发行，主编为周志华教授，执行主编为周昆教授，致力于为全球计算机领域学者提供一个高水平的交流平台，**是我国唯一的“计算机科学 理论与方法”方向SCI收录期刊**。



影响因子



特色栏目

观点

优秀青年科学家论坛

开源软件与数据集

智慧城市与城市计算

顶会论文投稿绿色通道

刊文领域

- 体系结构
- 理论计算机科学
- 图形图象
- 软件
- 网络及通信
- 信息安全
- 人工智能
- 信息系统
- 交叉领域

收录推荐

- CCF B
- SCI（E）、EI、 Scopus
- DBLP，INSPEC

卓越计划领军期刊

https://link.springer.com/journal/11704　　https://journal.hep.com.cn/fcs

Thank you!

郭兰哲

南京大学

 guolz@nju.edu.cn

 www.lamda.nju.edu.cn/guolz