SHERPA: A Model-Driven Framework for Large Language Model Execution

Boqi (Percy) Chen[1], Kua Chen[1], José Antonio Hernández López[2], Gunter Mussbacher[1], Dániel Varró[1,3], Amir Feizpour[4]

28th International Conference on Model Driven Engineering Languages and Systems
MODELS 25
5-10 October 2025
Grand Rapids, MI, USA

[1]Electrical and Computer Engineering, McGill University, Canada
[2]Department of Computer Science and Systems, University of Murcia, Spain
[3]Department of Computer and Information Science, Linköping University, Sweden
[4]Aggregate Intellect Inc., Canada

LLMs Are Powerful, But…

- Large language models (LLMs) are increasingly used in supporting various software engineering activities

LLMs Are Powerful, But…

- Large language models (LLMs) are increasingly used in supporting various software engineering activities
- LLMs are powerful, but they still have clear **limitations**

Miss **task context**

Handling **complexity**

Following **best practices**

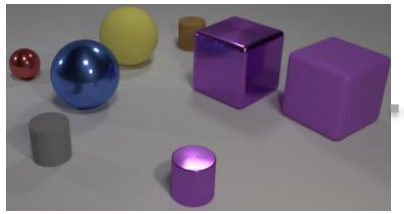
Ignore the *metamodel*

Fail to capture *dependency*

Violate *patterns/constraints*

In MDE tasks…

Example: Scene Graph QA



Scene graph

How many metal objects would there be if you didn't include spheres?

Input question

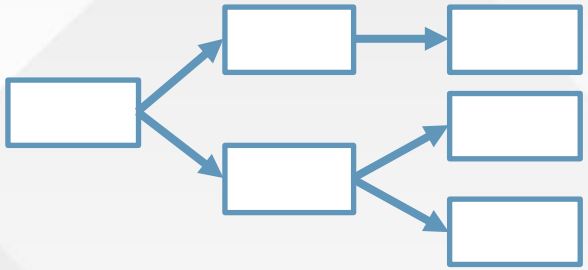
Observation: LLM handles different types of questions with different accuracy

How to model this information in the LLM task execution?

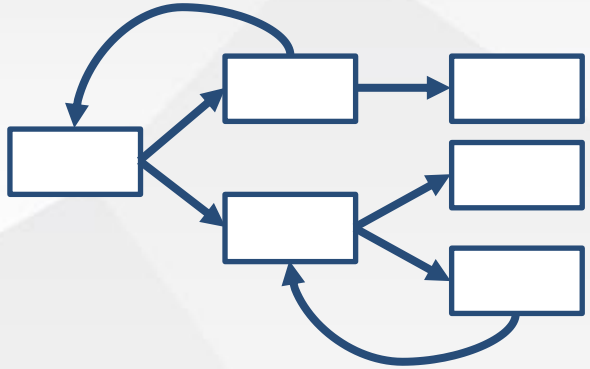
LLM Execution Workflow

Various **customized workflows** have been proposed to break down complex execution and capture human best practices

Chains



Trees



Simple Graphs

⚠ Steps **restricted** to nodes

⚠ No explicit **guard** conditions

⚠ Limited tool **compatibility** (e.g., reachability checks)

⚠ Tightly **coupled** to program logic

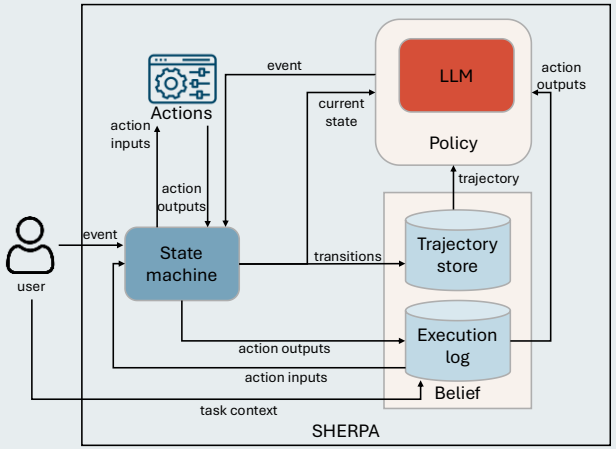
MDE to the Rescue

- ⚠ Steps restricted to nodes
- ⚠ No explicit guard conditions
- ⚠ Limited tool compatibility
- ⚠ Tightly coupled to program logic

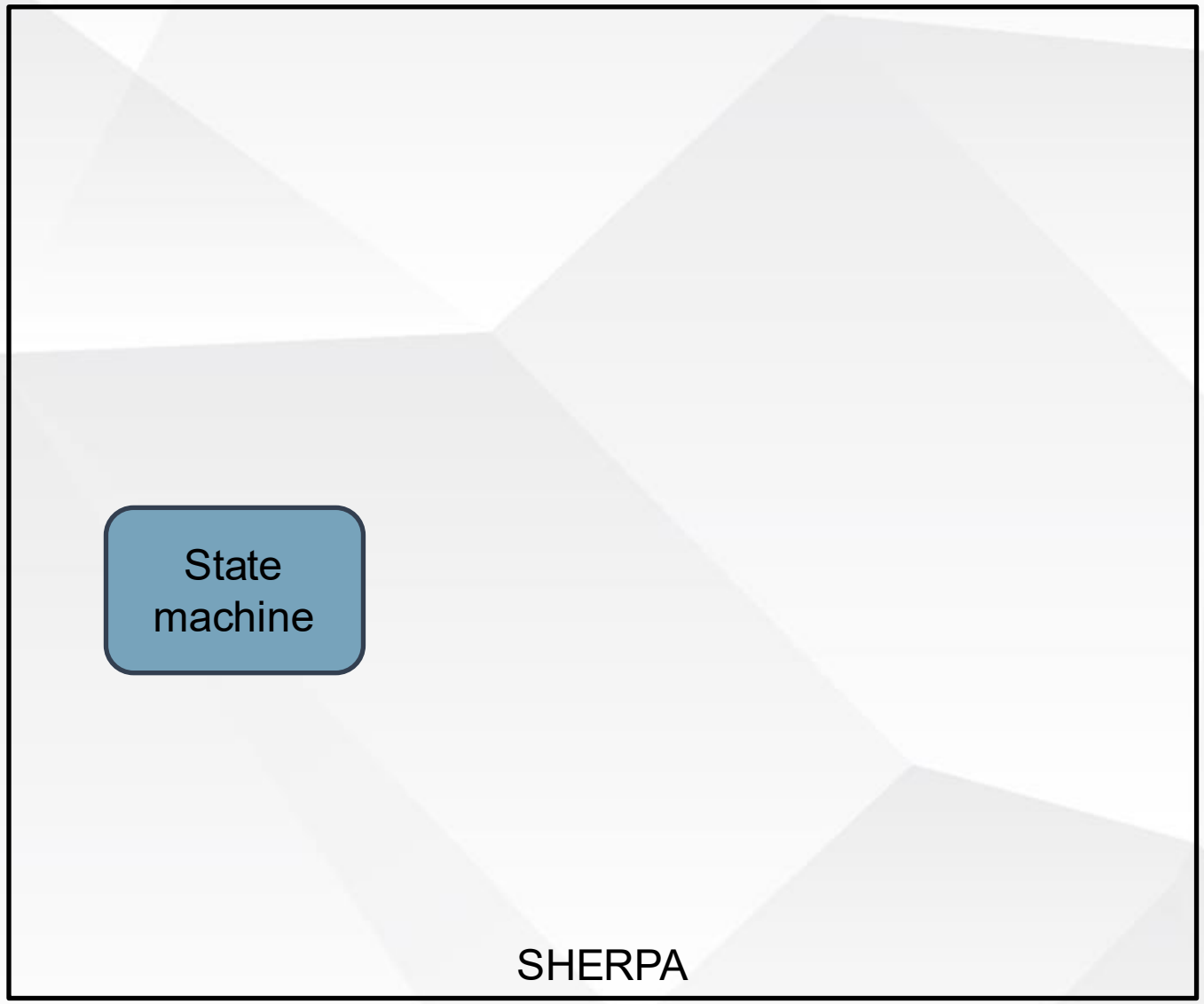
- 🧩 More **expressive** models
- 🧩 Explicit **configurations**
- 🧩 Rich **tool** supports
- 🧩 Strong **modularity**

In the MDE context…

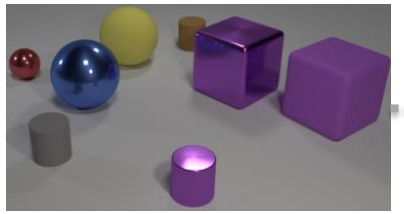
💡 State machine-based workflow (Sherpa)



Sherpa Architecture



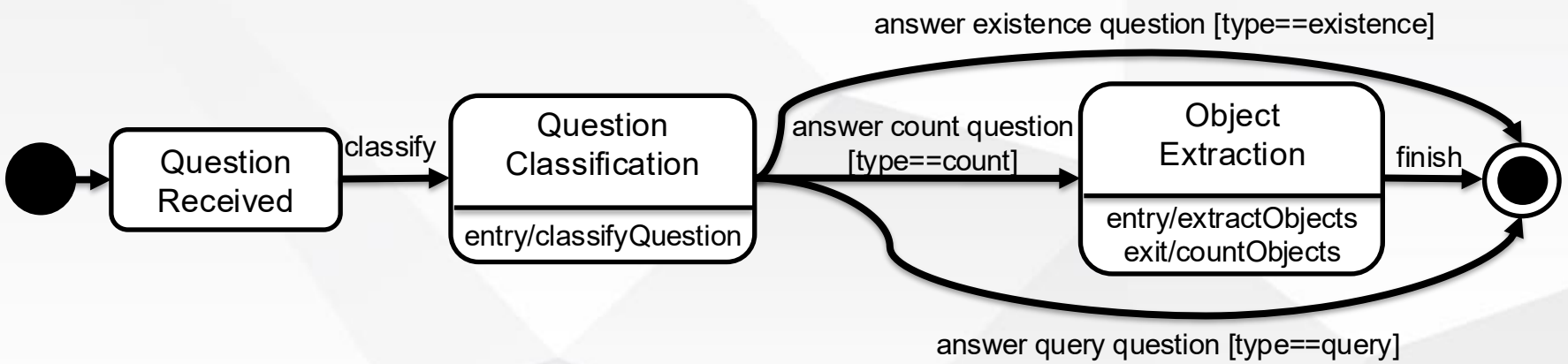
Example: Scene Graph QA



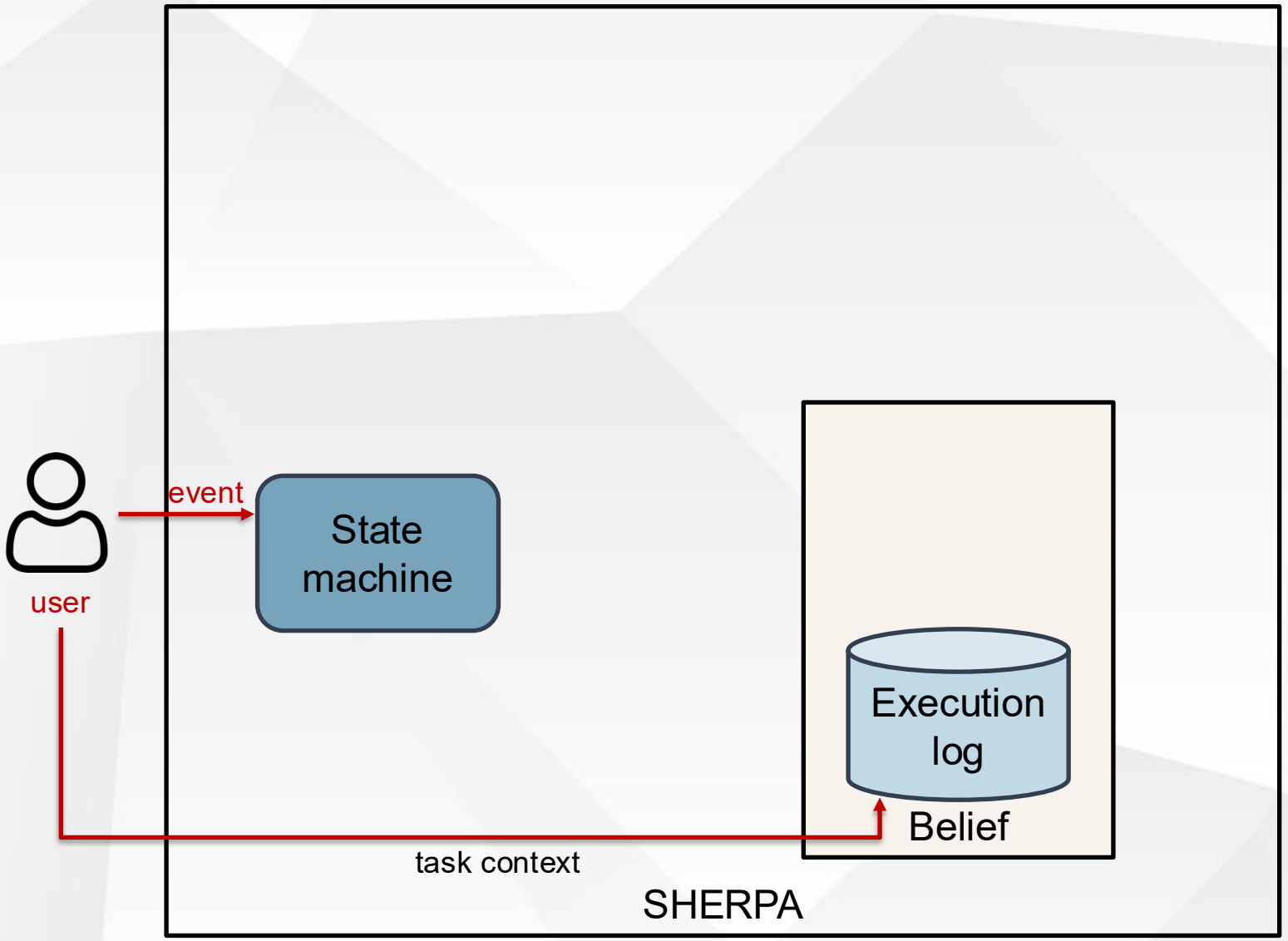
Scene graph

How many metal objects would there be if you didn't include spheres?

Input question

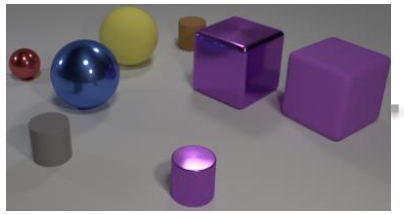


Sherpa Architecture



User triggers an **event** and provide **context**

Example: Scene Graph QA



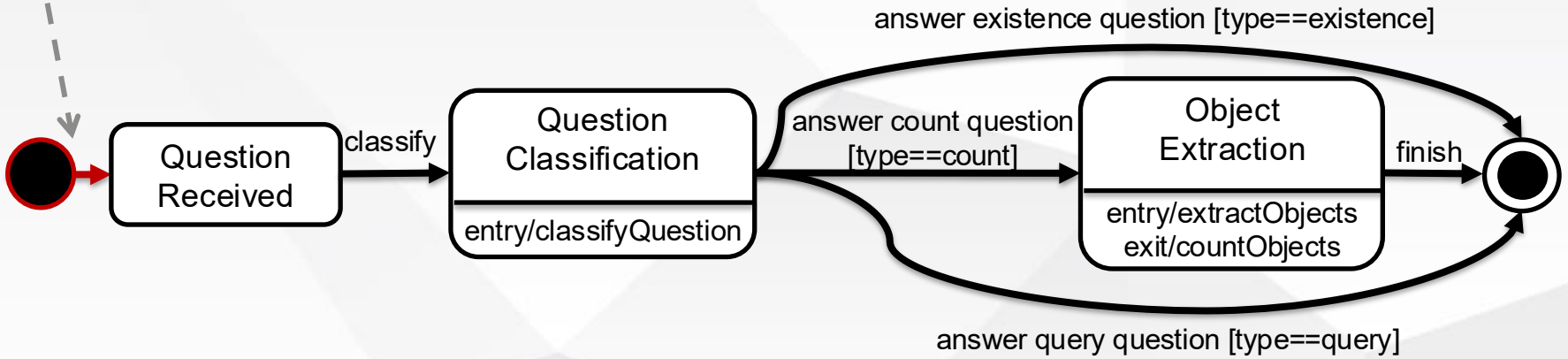
Scene graph

How many metal objects would there be if you didn't include spheres?

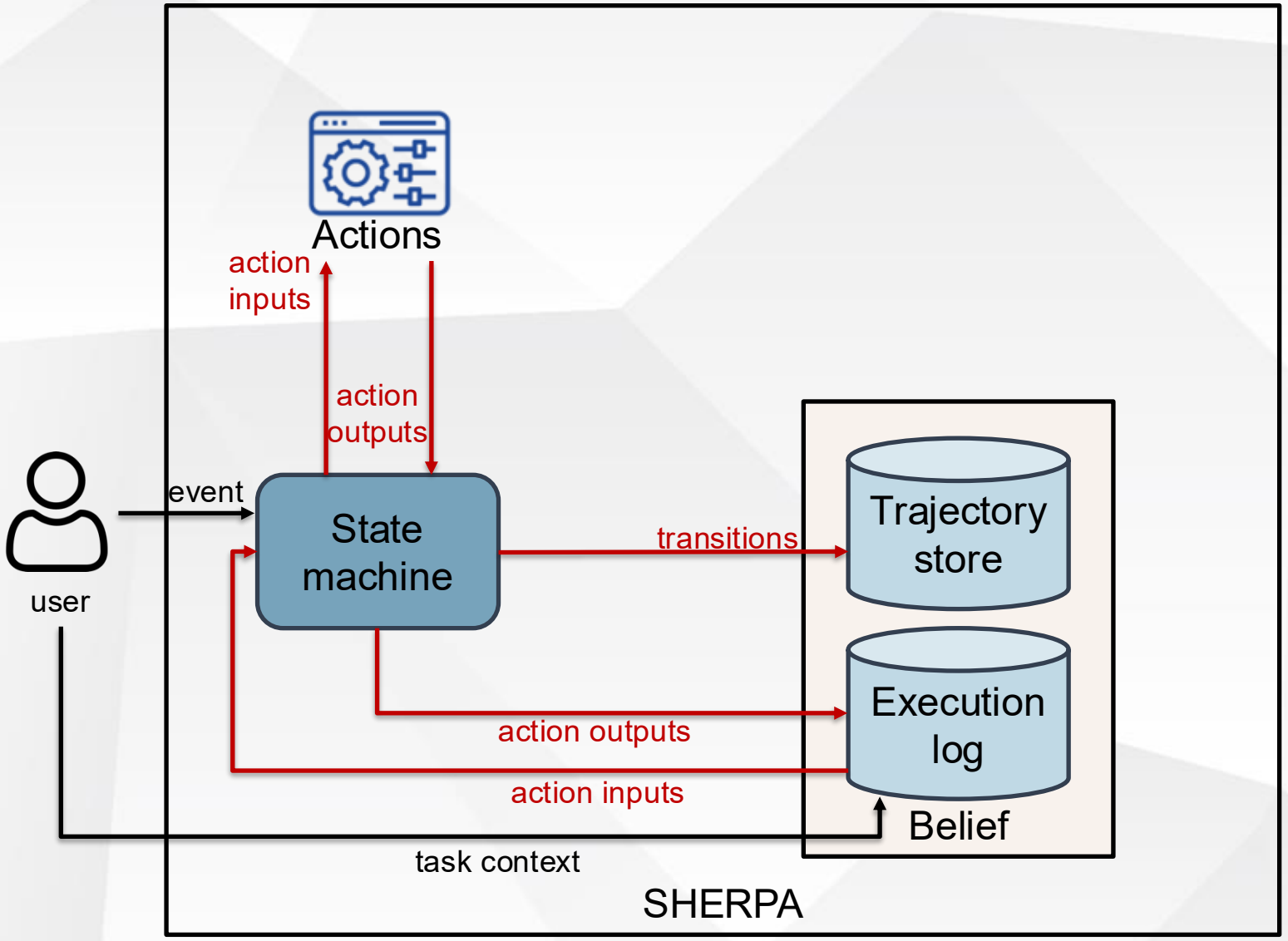
Input question

Execution trajectory

User triggers the initial transition



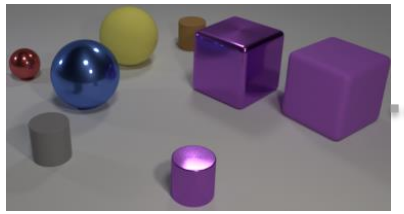
Sherpa Architecture



State machine:

- calls **actions** (in both *states* and *transitions*)
- reports **action results**
- record the **transition**

Example: Scene Graph QA



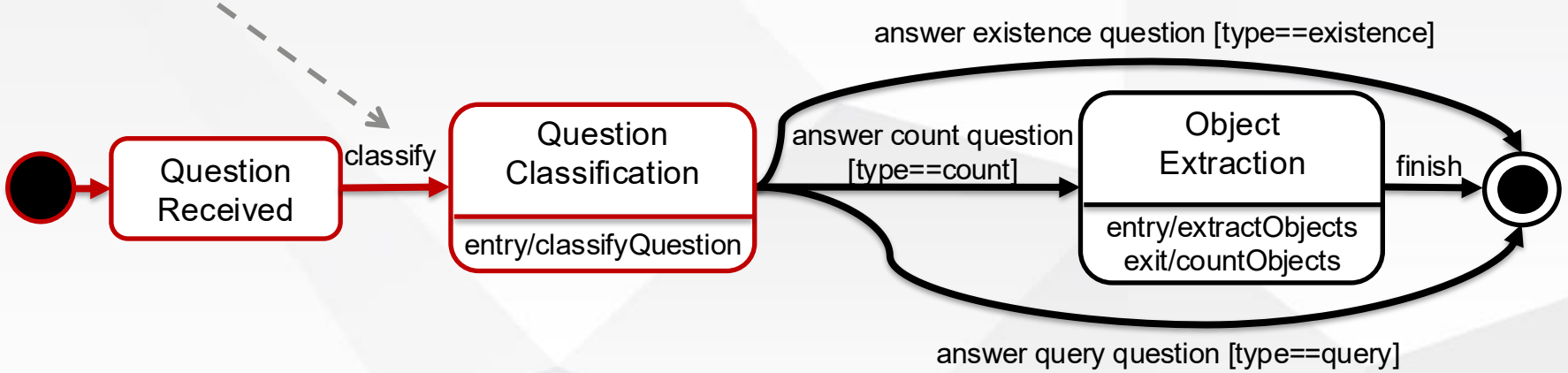
Scene graph

How many metal objects would there be if you didn't include spheres?

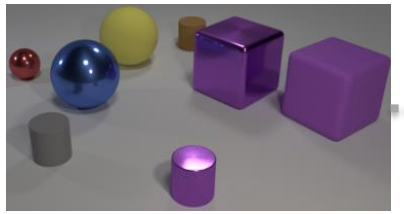
Input question

Execution trajectory

Continuing transition
(no decision-making)



Example: Scene Graph QA



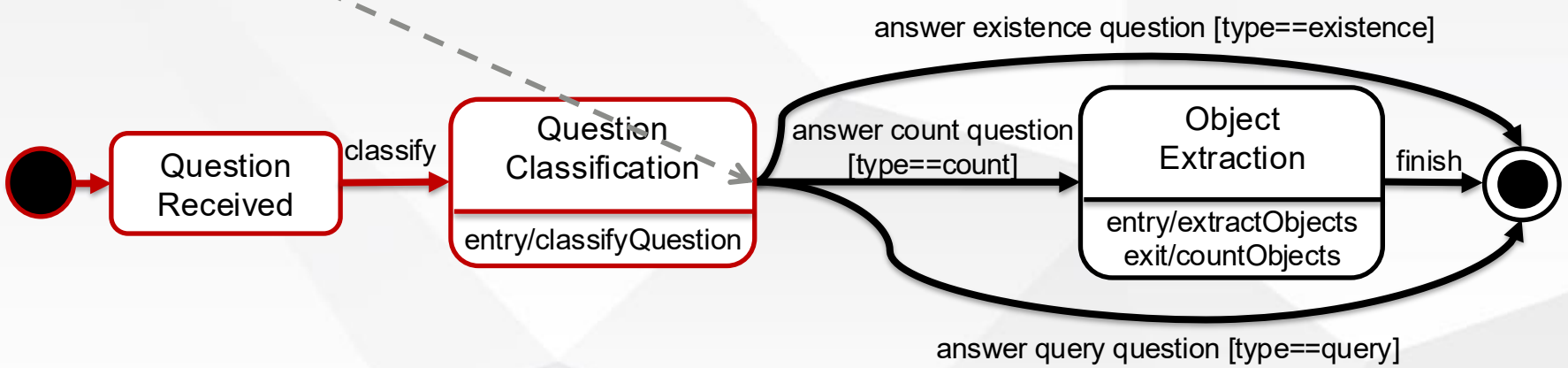
Scene graph

How many metal objects would there be if you didn't include spheres?

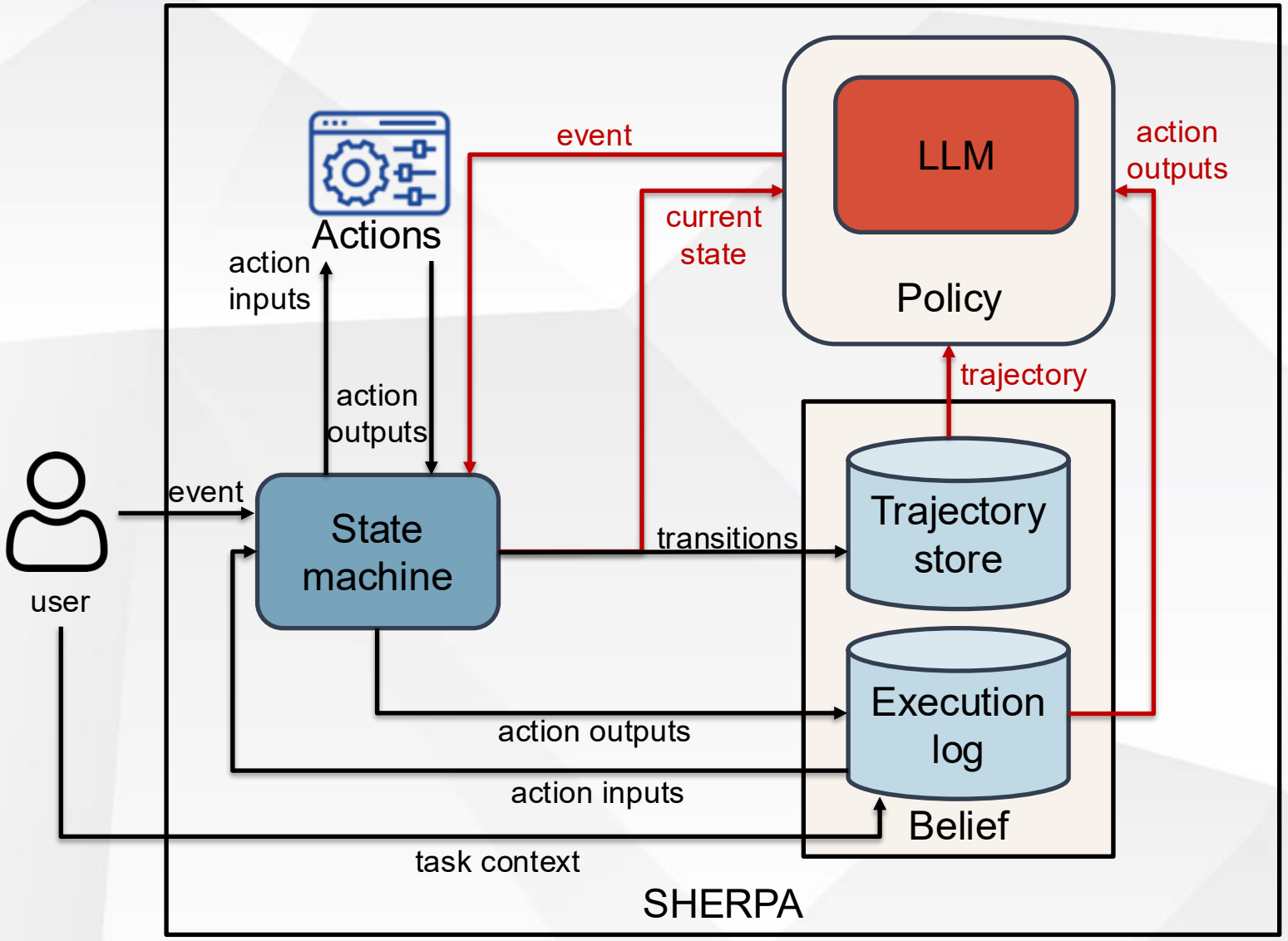
Input question

Execution trajectory

Branching



Sherpa Architecture



LLM-based policy selects the next **event** based on the **current state** and the **belief**

Transition Selection Prompt

```
You are a helpful assistant helping the user to answer a question…
```

Task description

```
Previous transitions…
Previous actions…
```

Execution history

```
You have a state machine to help you solve the task. You are currently in
the “routing” state…
```

Current state description

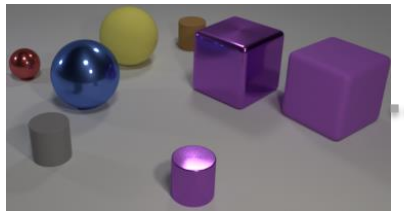
```
Available transition are:
{ “name”: “judge_question”,
    “description”: “Answer with yes/no, …”,
    “parameters”: []}
{  “name”: “count_question”,
    “description”: “Answer with a number…”
    “parameters”: [{…}, {…}]}
…
```

Available transitions

```
Select the best next action in the following JSON format…
```

Output instruction

Example: Scene Graph QA



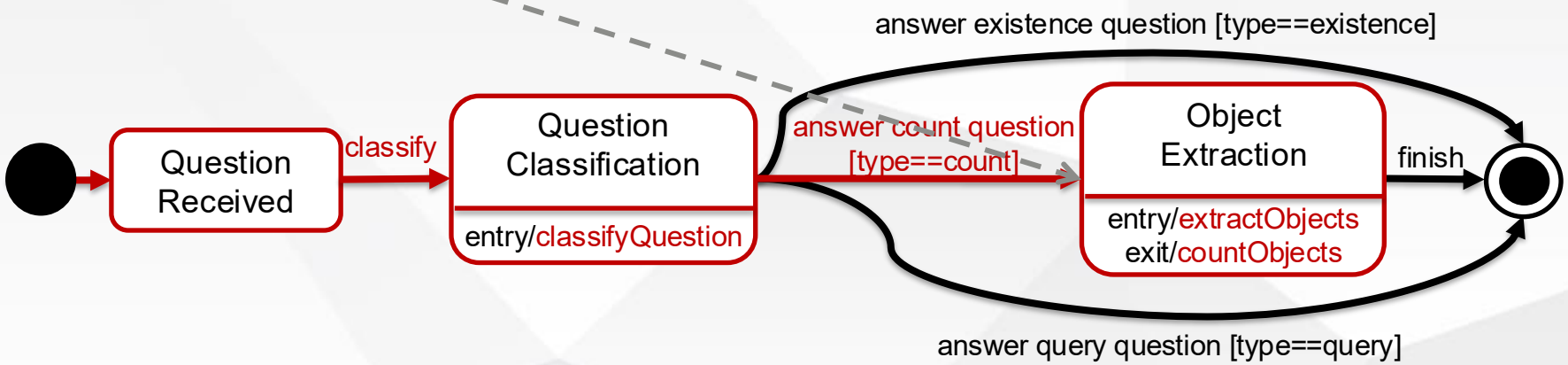
Scene graph

How many metal objects would there be if you didn't include spheres?

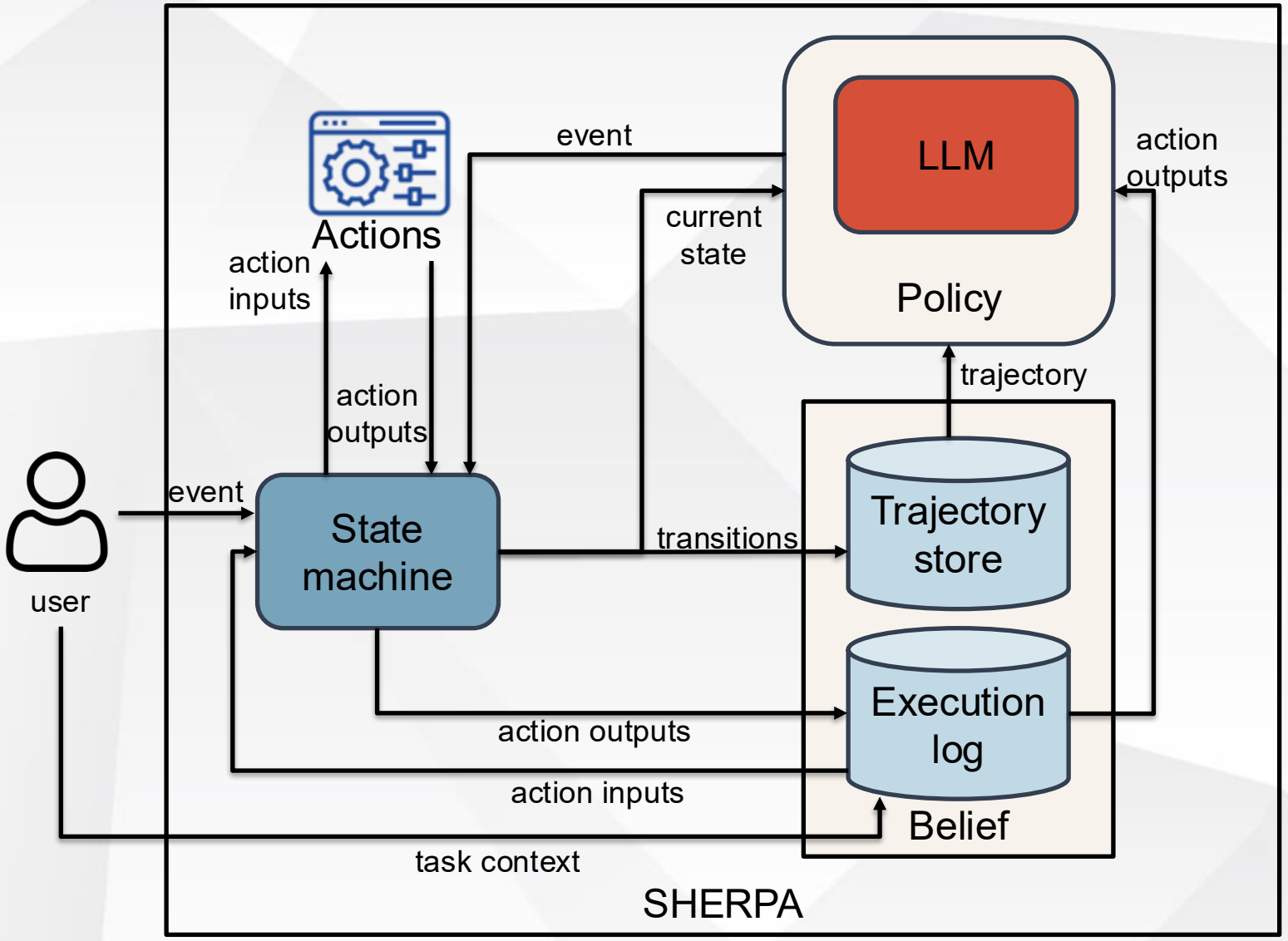
Input question

Execution trajectory

Classify as counting question



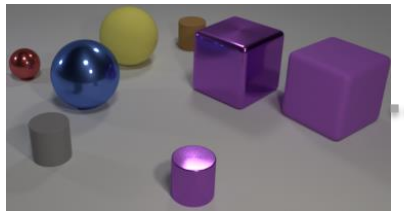
Sherpa Architecture



The process repeats until

- reaching **termination**
- exceeding **budget**
- encountering tags for **human help**

Example: Scene Graph QA



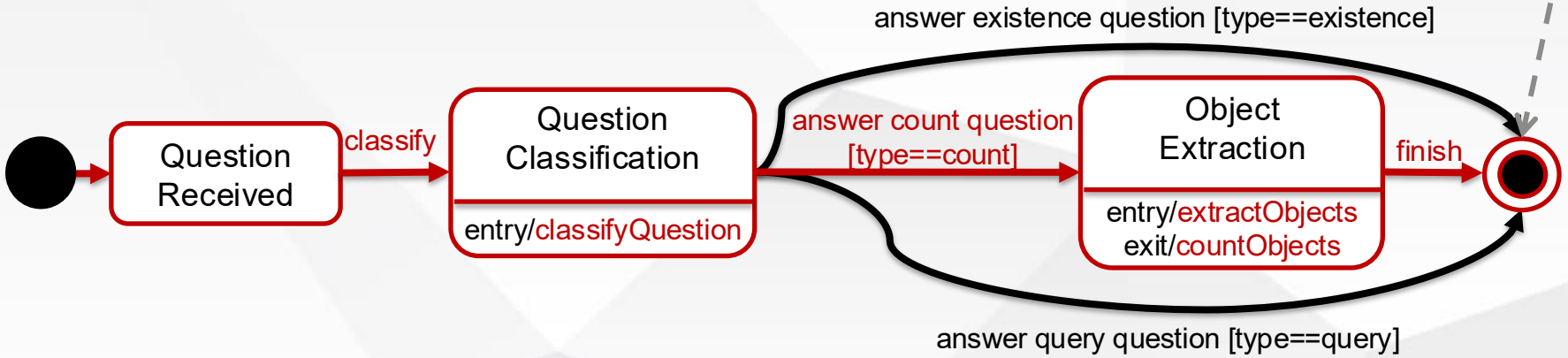
Scene graph

How many metal objects would there be if you didn't include spheres?

Input question

Execution trajectory

Terminate



Evaluation setup

Tasks

- Code generation
- Class name generation
- Scene-based QA

Metrics

- Pass@1
- F1 score
- Accuracy

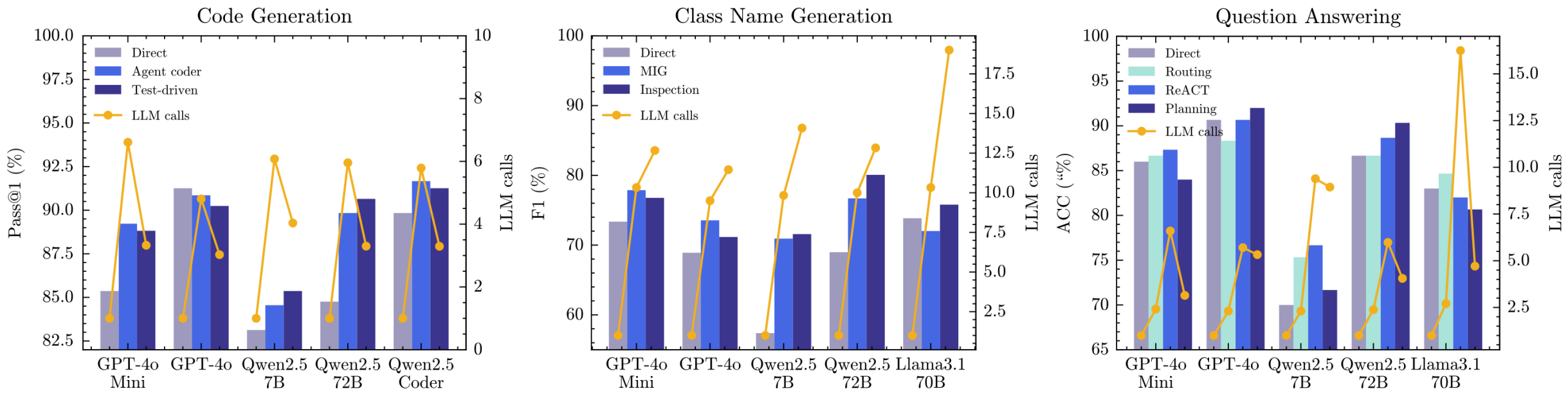
LLMs

- GPT-4o / GPT-4o mini
- Qwen 7B/72B
- *Qwen coder, Llama3.1 70b*

Compared approaches

- **Direct**ly using prompting
- Existing approaches
- Manual creation

Evaluation

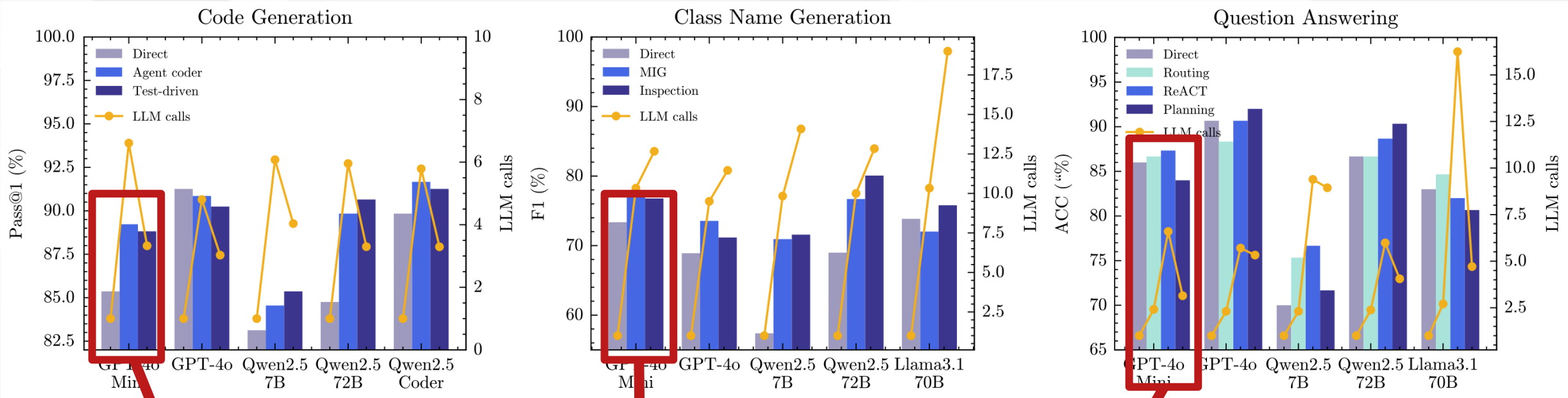


Performance of only using LLMs

Performance of different state machines

Number of LLM calls

Evaluation



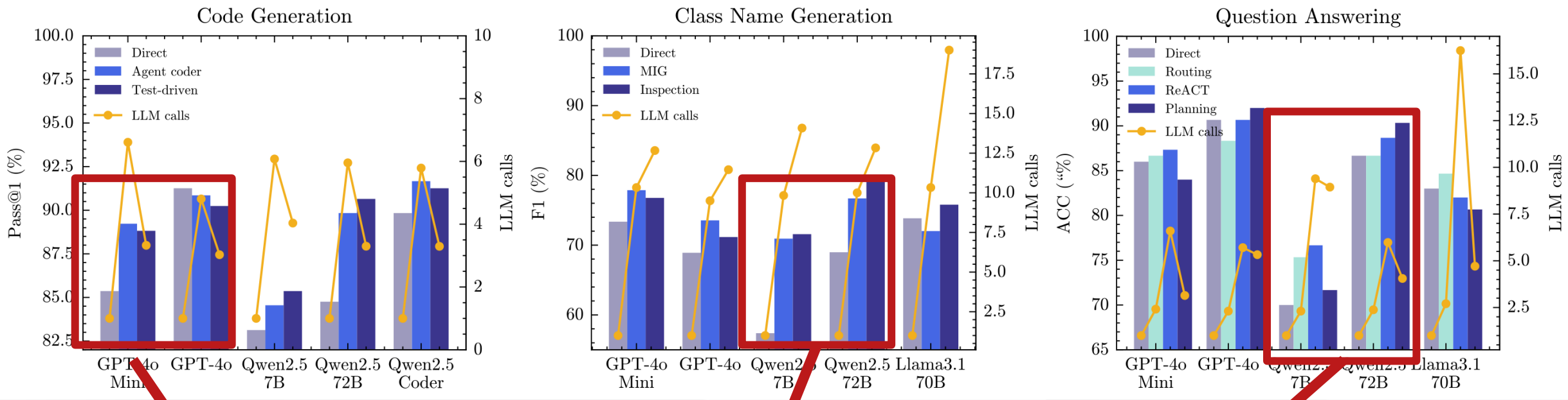
Performance of only using LLMs

Performance of different state machines

Number of LLM calls

Performance of state machines generally outperforms LLM-only approaches

Evaluation



Performance of only using LLMs

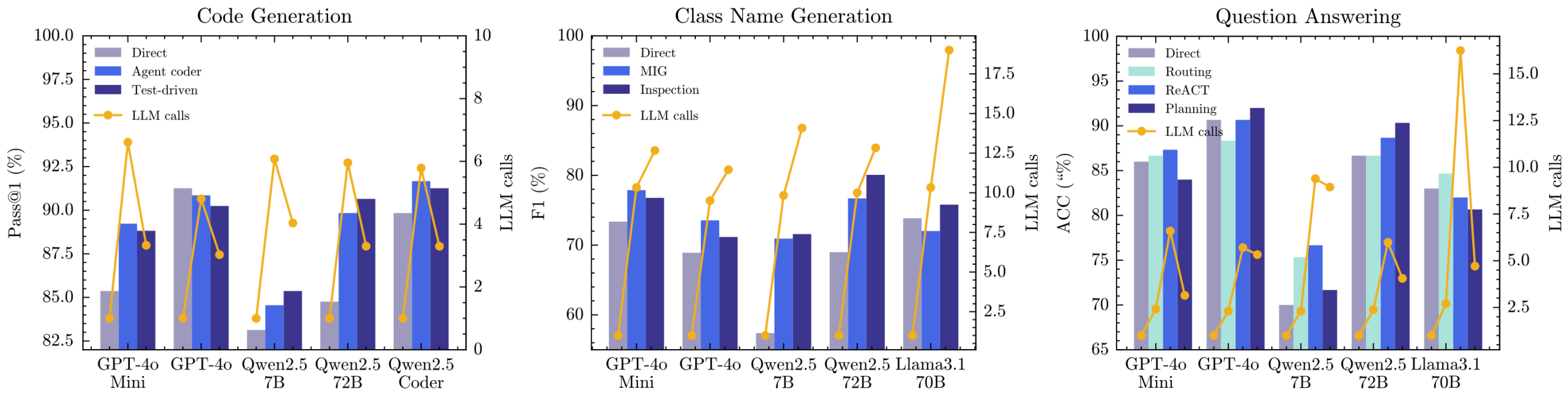
Performance of different state machines

Number of LLM calls

Performance of state machines generally outperforms LLM-only approaches

The effect is more profound on smaller LLMs

Evaluation



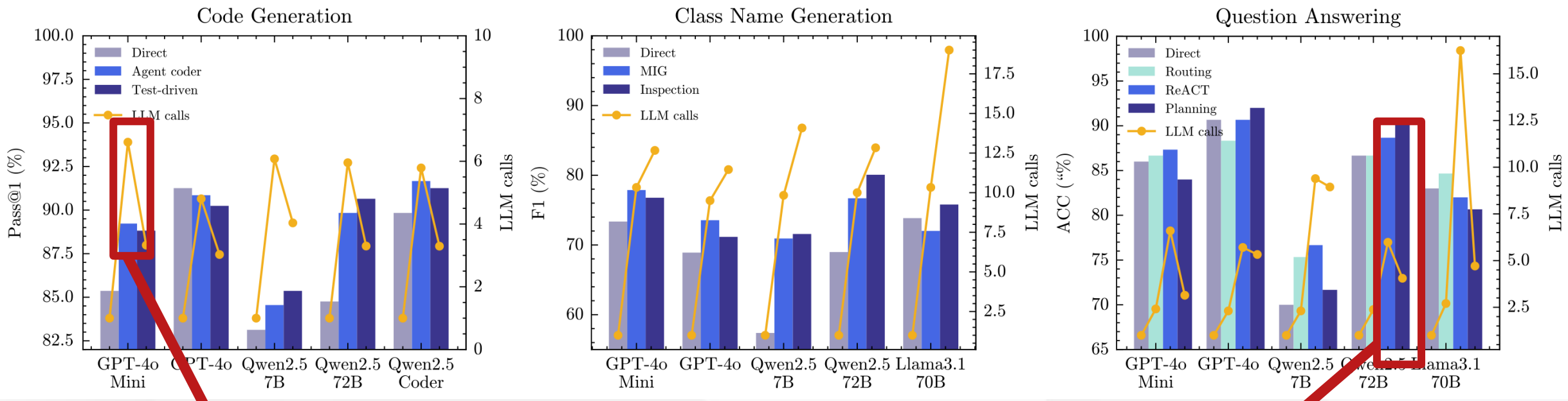
Performance of only using LLMs

Performance of different state machines

Number of LLM calls

Different state machine configurations significantly impact the performance

Evaluation



Performance of only using LLMs

Performance of different state machines

Number of LLM calls

Different LLM configurations significantly impact the performance

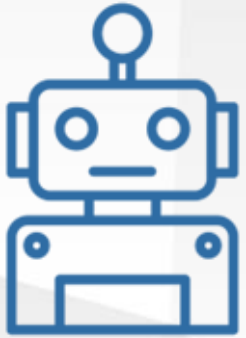
Optimizing state machines for efficiency reduces the number of LLM calls

Future work

Human-in-the-loop

Graphics Editors

Multi-agent support



RL-based policy

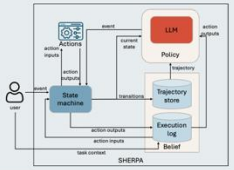
Thank you

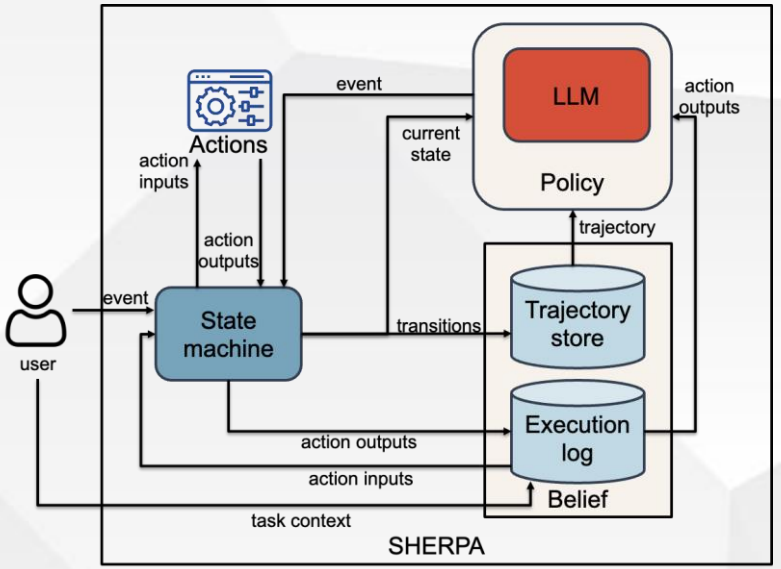
- ⚠ Steps restricted to nodes
- ⚠ No explicit guard conditions
- ⚠ Limited tool compatibility
- ⚠ Tightly coupled to program logic

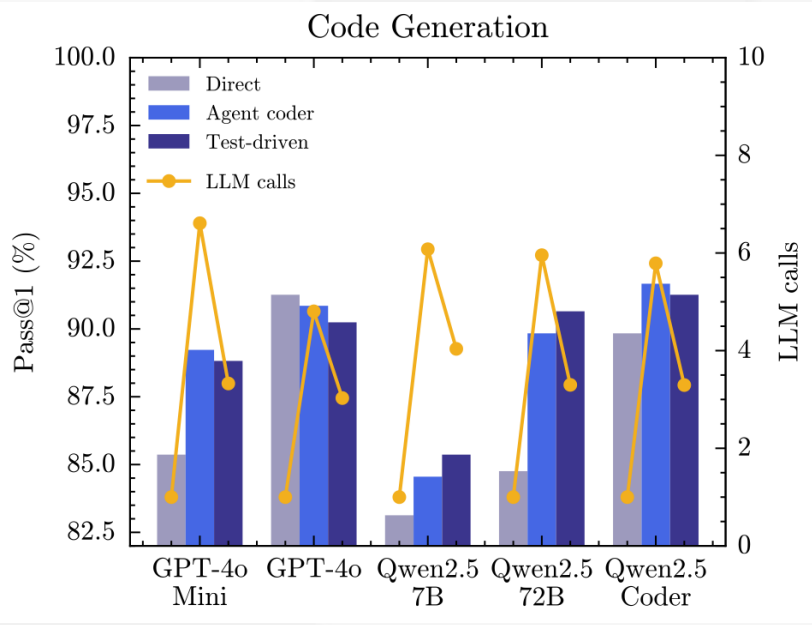
- 🧩 More **expressive** models
- 🧩 Explicit **configurations**
- 🧩 Rich **tool** supports
- 🧩 Strong **modularity**

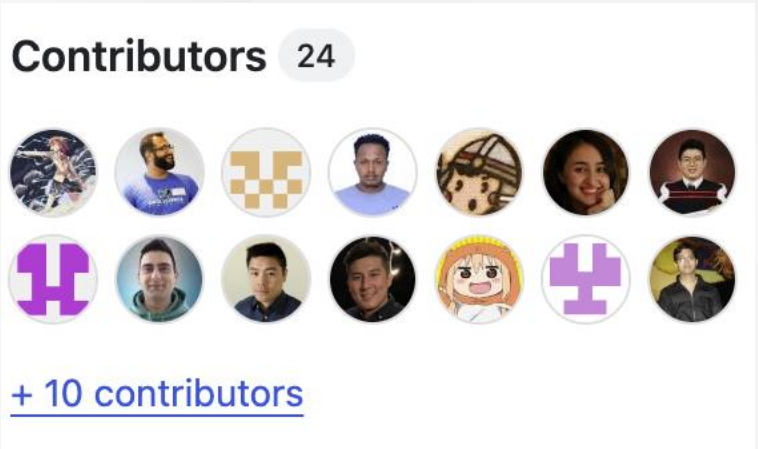
In the MDE context...

💡 State machine-based workflow (Sherpa)









Try Sherpa now!

```
pip install sherpa-ai
```

Project repository

Artifacts

Paper preprint