# VeriSafe Agent

## Safeguarding Mobile GUI Agent via Logic-based Action Verification

Jungjae Lee (KAIST)*, **Dongjae Lee (KAIST)***, Chihun Choi (Korea University), Youngmin Im (KAIST), Jaeyoung Wi (KAIST), Kihong Heo (KAIST), Sangeun Oh (Korea University), Sunjae Lee (Sungkyunkwan University), Insik Shin (KAIST)

KAIST KOREA UNIVERSITY 성균관대학교 SUNG KYUN KWAN UNIVERSITY

# Limitations of AI Agents: Low Reliability

AI · CODING

## An AI-powered coding tool wiped out a software company's database, then apologized for a 'catastrophic failure on my part'

BY **BEATRICE NOLAN**
TECH REPORTER

July 23, 2025 at 7:22 AM EDT

## McDonald's ends AI experiment after drive-thru ordering blunders

After working with IBM for three years to leverage **AI to take drive-thru orders**, McDonald's called the whole thing off in June 2024. The reason? A slew of social media videos showing confused and **frustrated customers** trying to get the AI to understand their orders.

## Lawyer Used ChatGPT In Court—And Cited Fake Cases. A Judge Is Considering Sanctions

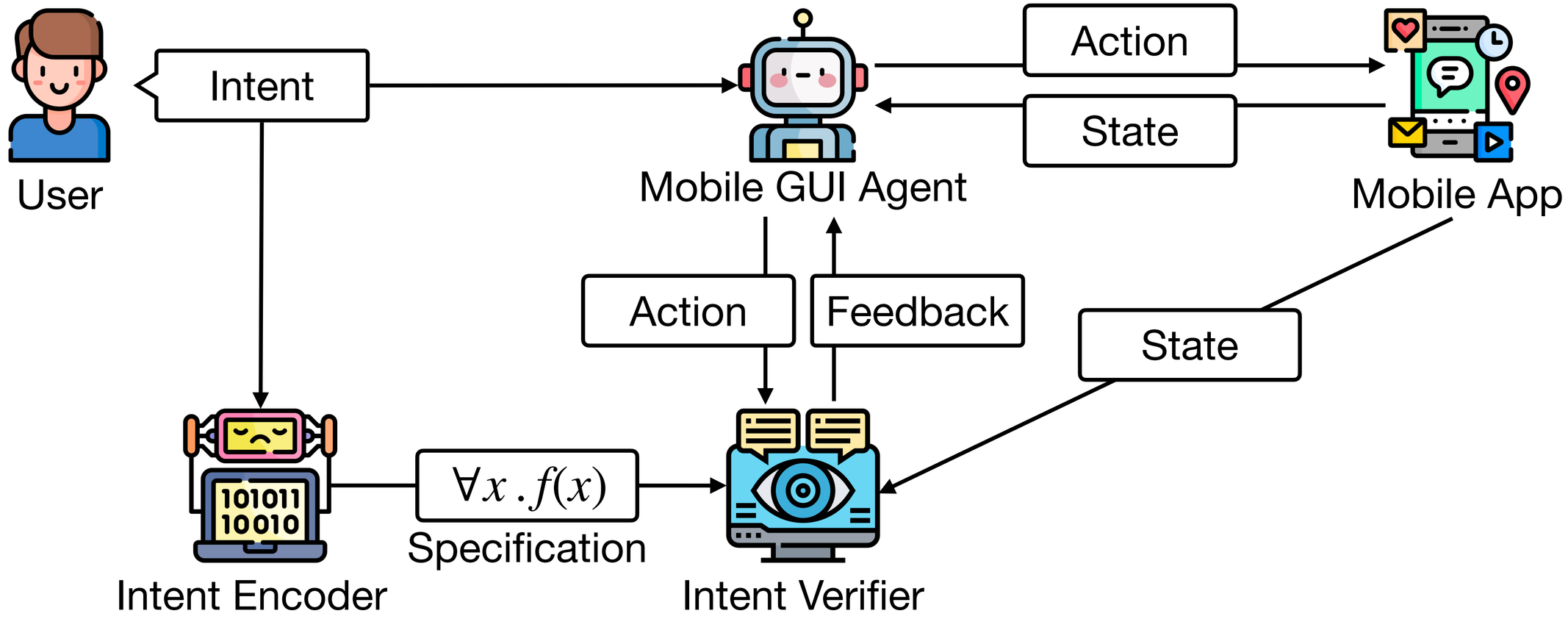
**By** Molly Bohannon, Former Staff. Molly Bohannon has been a Forbes news reporter since 2023.

Published Jun 08, 2023, 02:06pm EDT, Updated Jun 08, 2023, 03:42pm EDT




- Molly Bohannon, 2023, Lawyer Used ChatGPT In Court—And Cited Fake Cases. A Judge Is Considering Sanctions
- Thor Olavsrud, 2025, 11 famous AI disasters
- Bertice Nolan, 2025, An AI-powered coding tool wiped out a software company's database, then apologized for a 'catastrophic failure on my part'

# Solution: Verify Agents' Action with Logic



# Solution: Verify Agents' Action with Logic

- **Verification Accuracy 38.2%** 
- **Success Rate in Complex Tasks 130%** 
- **Cost 96%** 

# Example: Booking a flight ticket

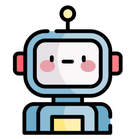
User

Book me a flight from ***Seoul*** to ***Hong Kong*** departing on ***Nov. 4th*** and returning on ***Nov. 9th***.


Intent Encoder

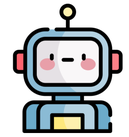
$$Ticket(\textit{from} = \textbf{"Seoul"}, \textit{to} = \textbf{"Hong Kong"}, \textit{depart} = \textbf{Nov. 4th}, \textit{return} = \textbf{Nov. 9th}) \Rightarrow Book$$

OK



Mobile GUI Agent

Select ***Nov. 4th*** from the calendar



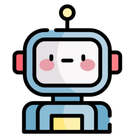
Mobile GUI Agent

Intent Verifier

$$Ticket(\textit{depart} = \textbf{Nov. 4th}) \Rightarrow OK$$

# Example: Booking a flight ticket

Select ***Nov. 11th*** from the calendar



Mobile GUI Agent

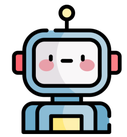
$Ticket(return = \textbf{\textit{Nov. 11th}}) \Rightarrow ERROR$

Intent Verifier

Feedback: "Return date should be ***Nov. 9th***, but found ***Nov. 11th***"

Intent Verifier

Select ***Nov. 9th*** from the calendar

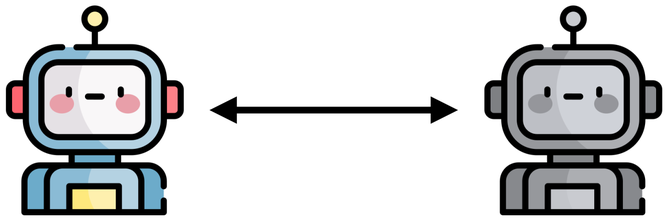


Mobile GUI Agent

$Ticket(return = \textbf{\textit{Nov. 9th}}) \Rightarrow OK$

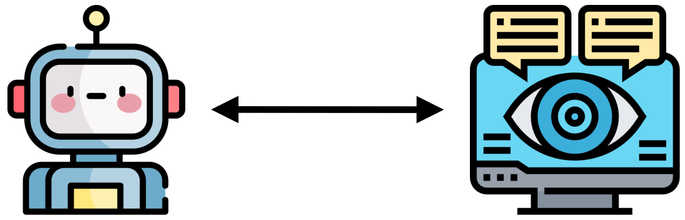
Intent Verifier

# VeriSafe Agent VS. Self-Reflection



Mobile Agent Verification Model

**Self-Reflection**



Mobile Agent Logical Verifier

**VeriSafe Agent**

| Advantages | Self-Reflection | VeriSafe Agent |
|---|---|---|
| 1. No hallucination during verification | X | O |
| 2. Consistent verification result | X | O |
| 3. Mathematically infer causes of errors | X | O |
| 4. No accuracy decline as task length increases | X | O |
| 5. Low cost and latency | X | O |

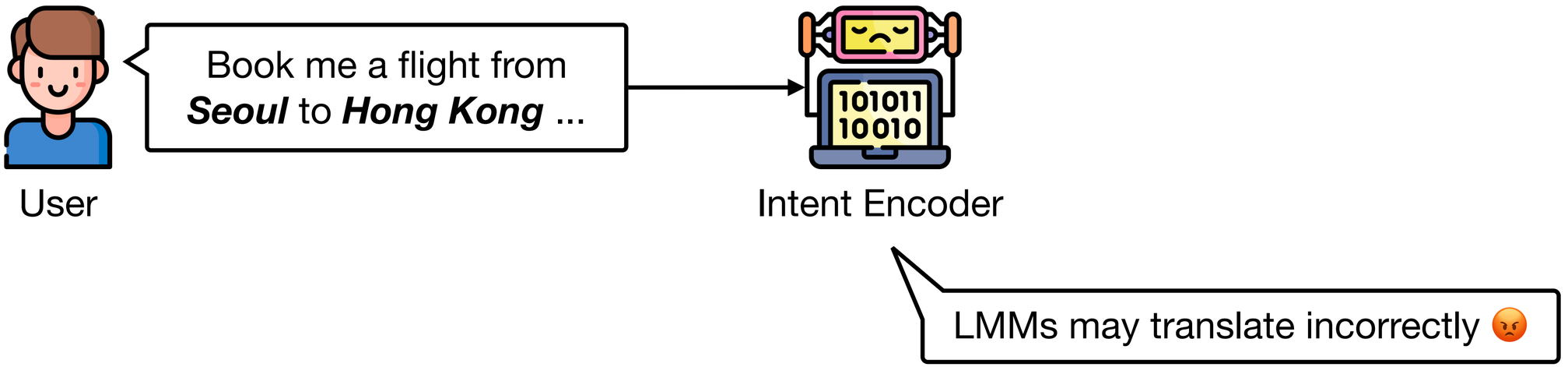
# Challenges to implement VeriSafe Agent

1. How can we **mitigate hallucinations** at intent encoding?

2. Which language is **the best for specification**?

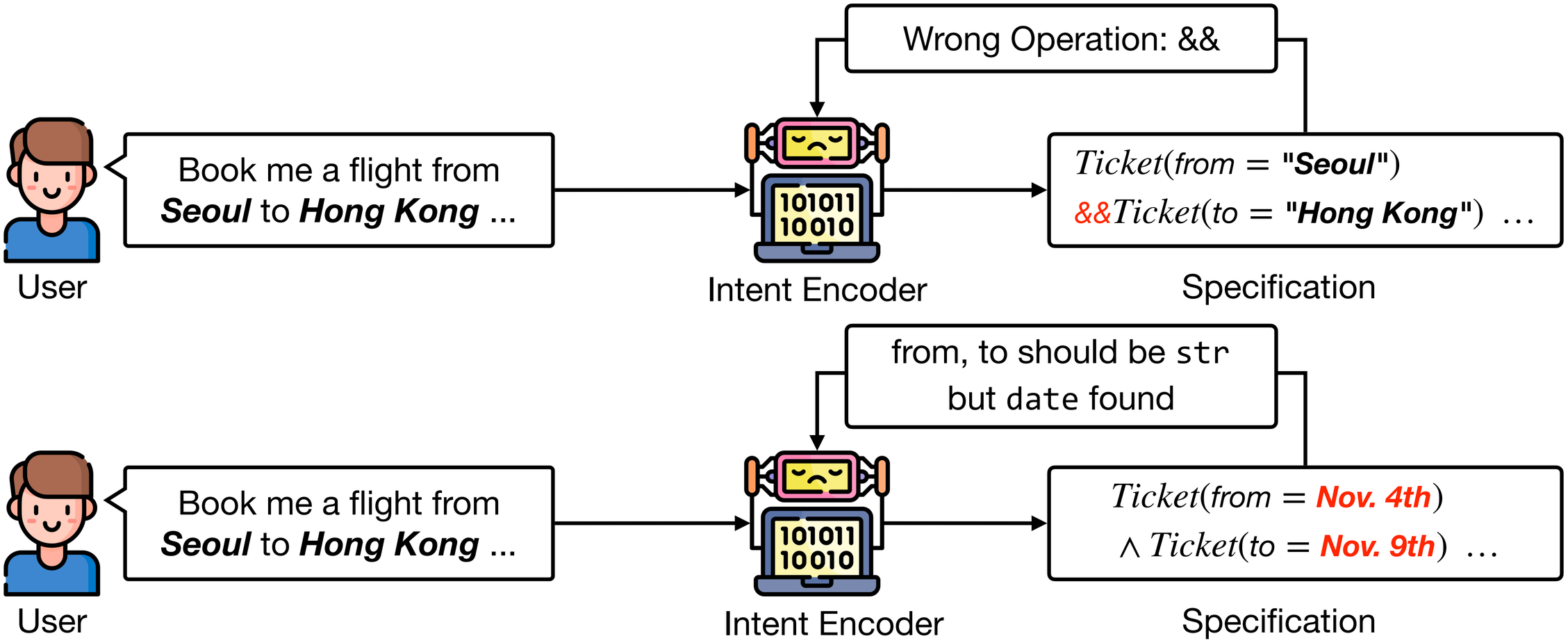
3. How do we implement **pre-action** verification?

# Challenge 1: Fallible Intent Encoding



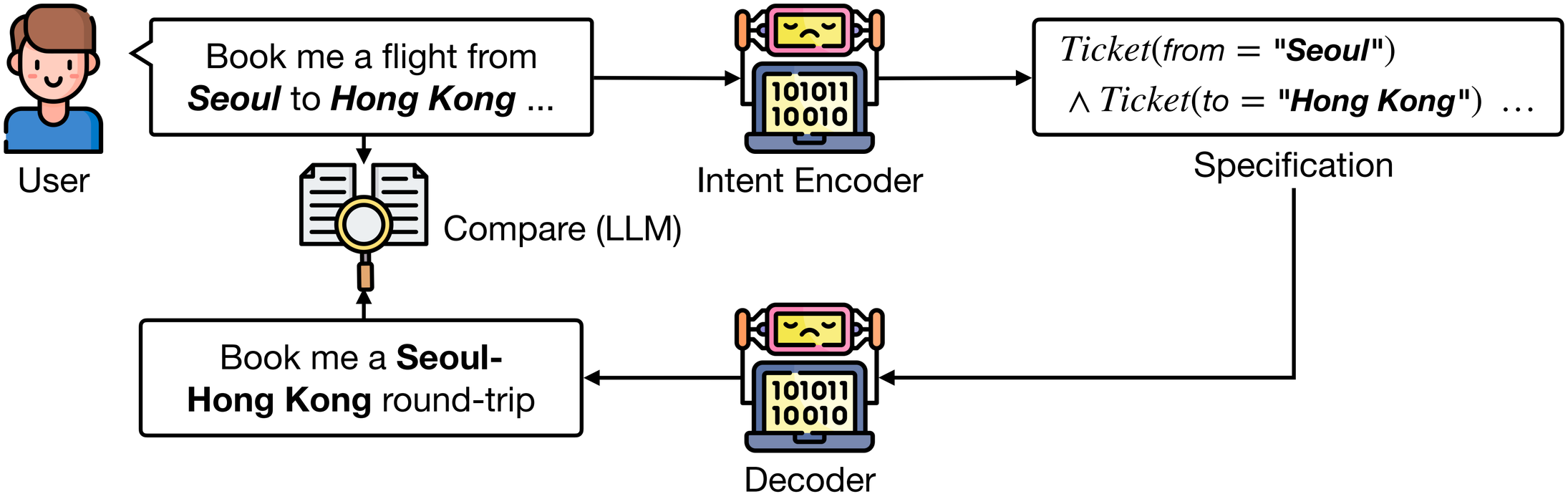
# Solution: Syntax & Type Check

- Check generated specification satisfies syntax and type constraints



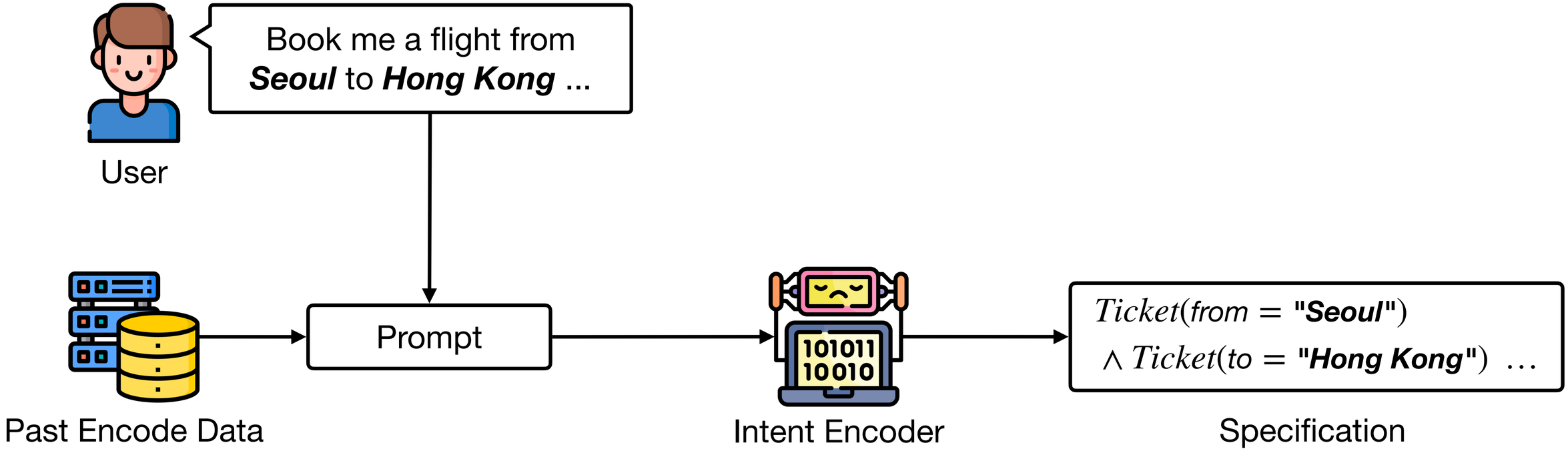
# Solution: Consistency Check

- Decode a logical formula into natural language, and compare with user intent
  - Decode(Encode("User Intent")) = "User Intent"
  - Minimizing missing or incorrect content



# Solution: Memory System

- Utilizing past encoding data as reference material

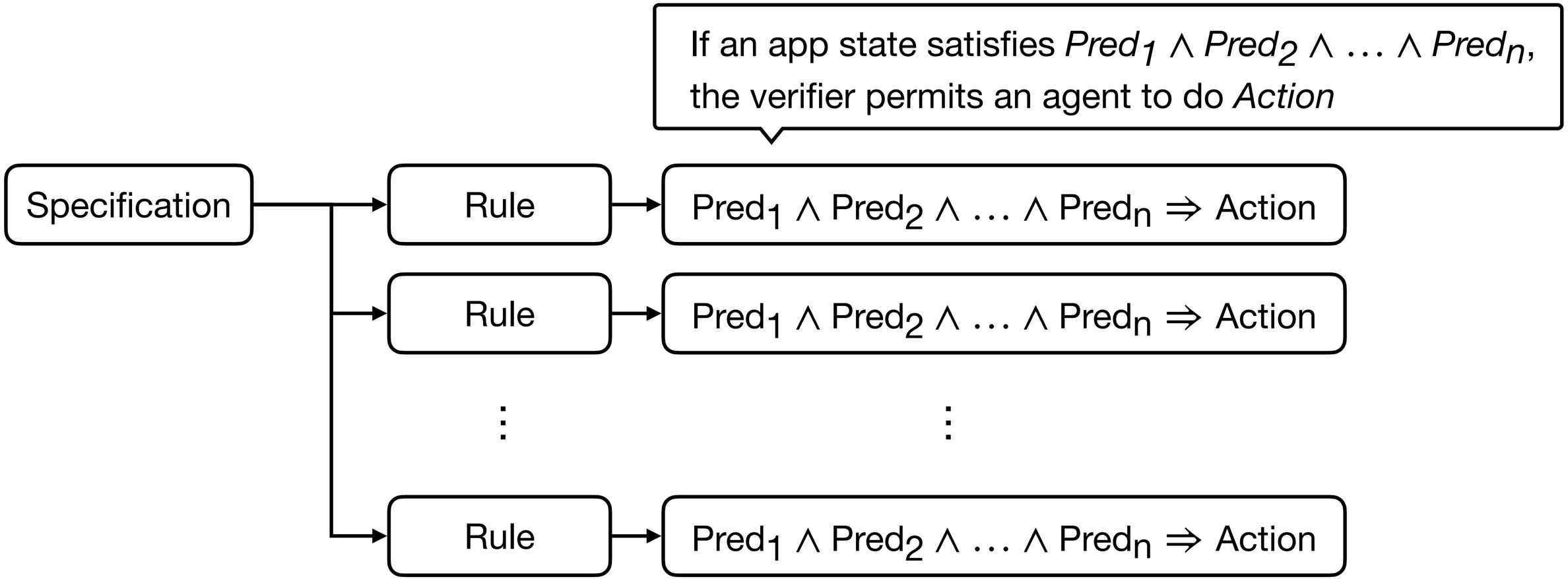


# Challenge 2: Language for Specification

- Full-featured language (e.g., first-order logic) is highly expressive
  - But, inefficient for writing specifications
  - Language is not app-optimized
- Properties that the optimal language must satisfy
  1. **Highly Expressive**: express most user intent or app states.
  2. **Easy to Use**: concise and intuitive formulation of user intent

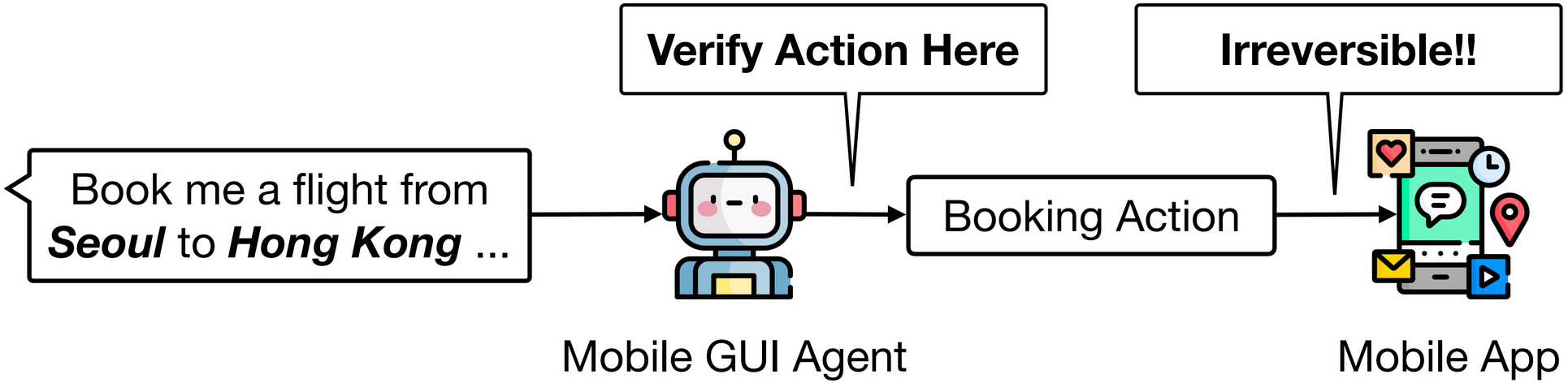
# Solution: Domain-Specific Language

- Inspired by rule-based programming language (e.g., Datalog)



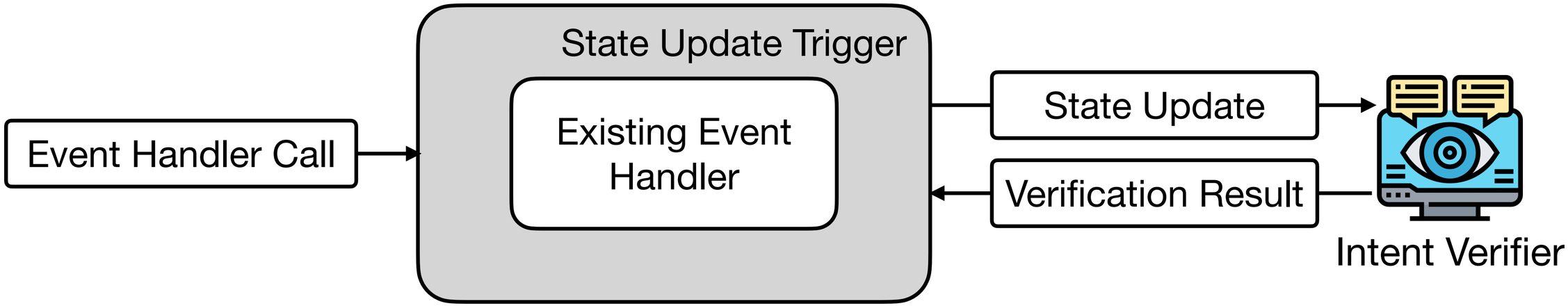
# Challenge 3: Pre-action Verification

- Mobile agents' action can be irreversible
  - Bank deposits, ticket purchases, payments, ...
- A verifier must confirm whether an action is correct before performing it.



# Solution: Developer Library and Simulation

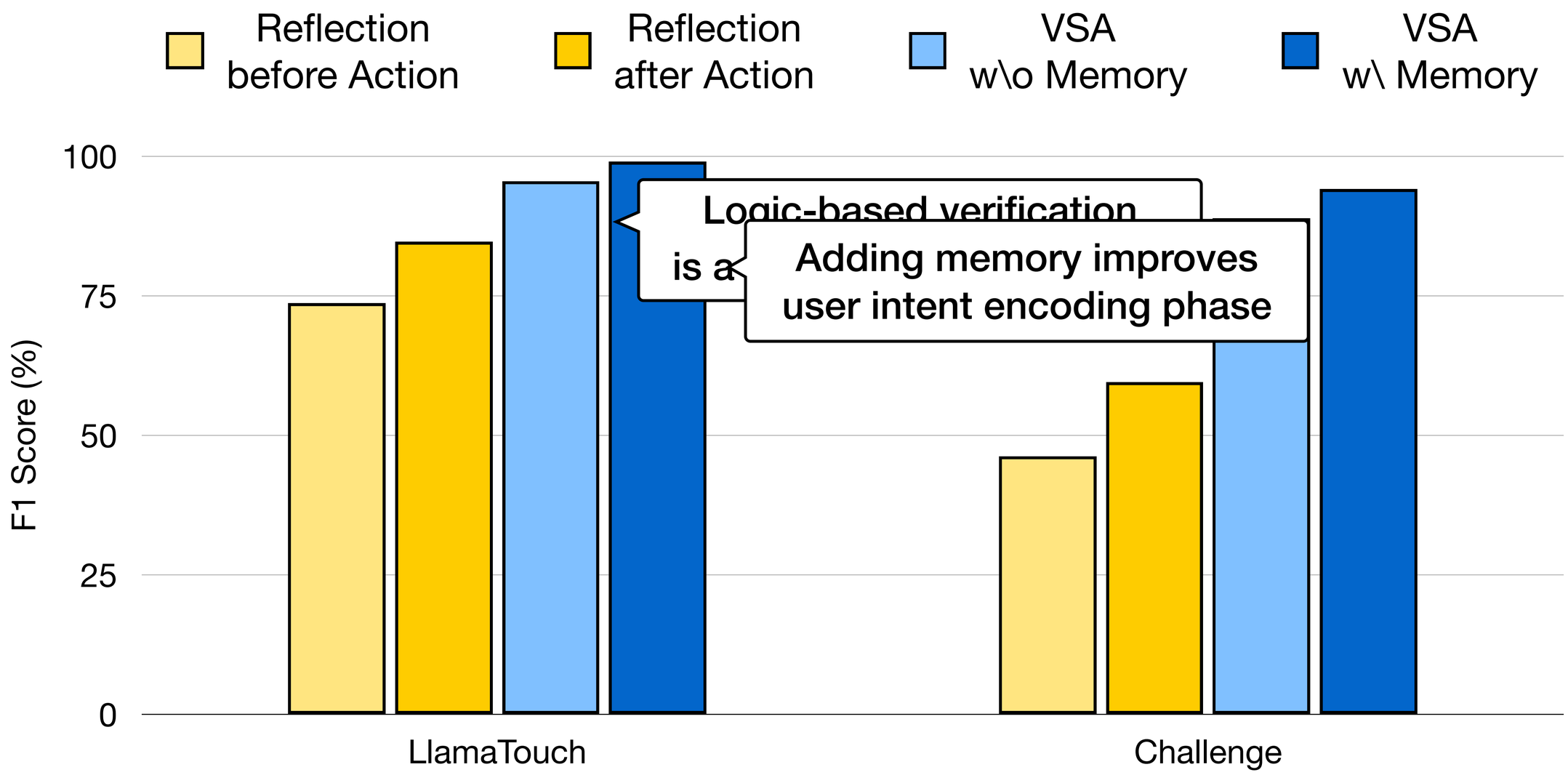
- App Developers insert state update triggers to existing event handlers.
- Trigger intercepts an event handler and simulates a state update result.
- If it violates a specification, the state update trigger stops state update.



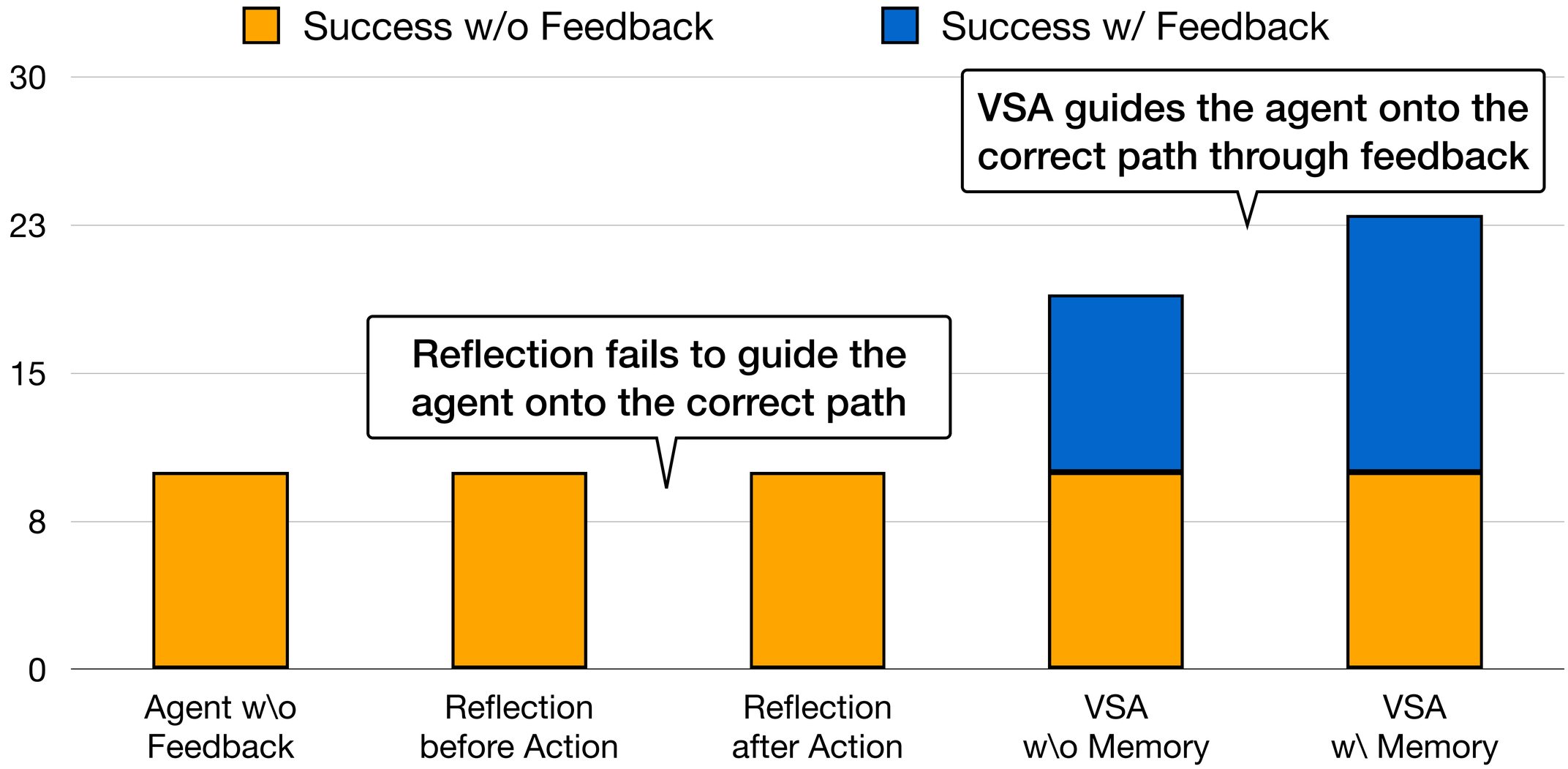
# Evaluation

- Benchmark
  - Simple tasks on mobile applications from LlamaTouch (125)
  - Custom-built challenge tasks (25)
- Model
  - GPT-4o
- Baseline
  - Self-reflection-based agent

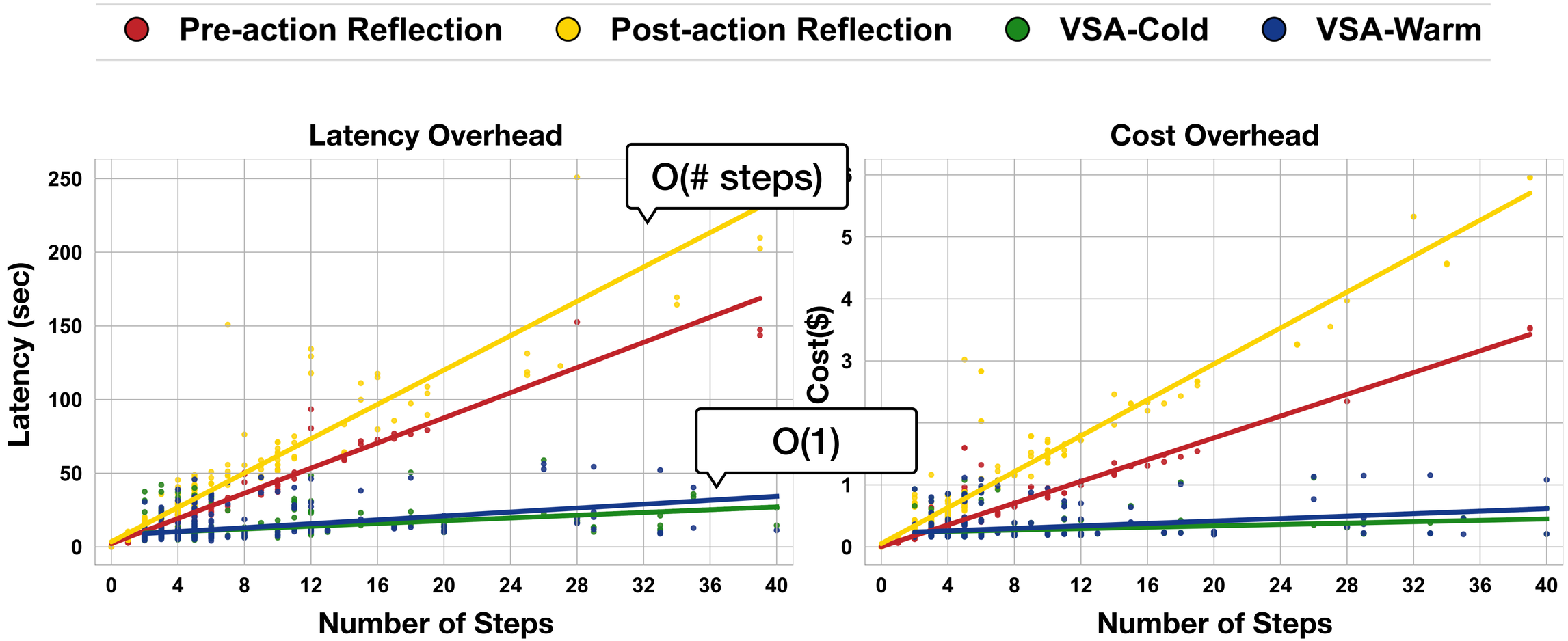
# Verification F1 Score



# Effectiveness of Feedback (Challenge)



# Latency and API Cost



# Summary

- Logic-based agent action verification **VeriSafe Agent**
- Outperforms self-reflection-based agent in both performance and cost.
- Contribution
  - Implement a trustworthy intent encoder and rule-based action verifier
  - Define app-optimized domain-specific language
  - Develop a developer library for pre-action verification
- Contacts: Dongjae Lee (dongjae.lee@prosys.kaist.ac.kr)