# Thinking Fast and Slow in AI

A cognitive architecture to augment both AI and human reasoning

Francesca Rossi

IBM Research

# A Multi-Disciplinary Team

AI (ML, KR, planning)
Software Engineering
Mathematics
Philosophy
Cognitive Science

Francesca Rossi, Murray Campbell, Lior Horesh, Keerthiram Murugesan – IBM Research, USA

AI, ML, cognitive architectures – SOFAI architecture and its instances, FASCAI

Andrea Loreggia – University of Brescia, Italy

Implementation, experiments – SOFAI for grid navigation

Francesco Fabiano – New Mexico State Univ., USA

Theoretical and experimental/implementation work -- SOFAI for planning

Nick Mattei – Tulane Univ., USA

Preferences, ML, data analysis

Biplav Srivastava – Univ. of South Carolina, USA

Planning – SOFAI, planning instance

Brent Venable – Univ. West Florida and IHMC, USA

Preferences, MDFT -- SOFAI

Marianna Bergamaschi Ganapini – Union College, NY, USA

Metacognition – SOFAI architecture, FASCAI

Vishal Pallagani – Univ. of South Carolina, USA and IBM Intern

Planning – Plansformer, SOFAI for planning, SOFAI for LLM/AI reasoning

Vedant Khandelwal – Univ. of South Carolina, USA

SOFAI for graph coloring

(late) Daniel Kanheman -- collaborator

# Talk outline

- Reasoning
- Neuro-symbolic AI
- Thinking fast and slow
- The basic SOFAI architecture
  - Planning
  - Grid navigation
- The generalized SOFAI architecture
  - Graph coloring
- AI-supported and nudged human decision making

## Reasoning

A process

- “the **process** of thinking about something in a **logical way** in order to form a conclusion or judgment” (Britannica)
- “the drawing of inferences or conclusions **through** the use of **reason**” (Merriam-Webster)

**Not** just the **output** of that **process**

”reason” and “logical way” are human-related concepts -- human system 2

## Reasoning tasks and beyond

### Reasoning tasks

- Humans need System 2 (reasoning) to solve them
  - Ex. Complex arithmetic
- Possibly in combination with experience (System 1)
  - Ex.: language translation

### Non-reasoning tasks

- Humans just use System 1
  - Ex.: face recognition

# Reasoners and reasoner simulators

A reasoner performs the reasoning process

For each input, it generates output using that process

Usually static, does not need data

Ex.: symbolic theorem prover, planner

Evaluation: do they use correctly the reasoning rules?

Reasoner simulator: a system that approximates the input/output behavior of a reasoner

For reasoning tasks

Ex.: LLM

Reasoners do not need data to do their job

Evaluation: do they use the reasoning rules?

Reasoner simulators exploit data/examples to be good simulators of the I/O behavior of a reasoner

But many reasoning tasks have an infinite space, so data is not enough to be a perfect simulator

Evaluation: test on datasets for human reasoning tasks

## Can ML/LLM reason?

- It **depends** on the **definition** of **reasoning** people use
  - **Yes**, because they **perform well** on **reasoning tasks**
    - Focus on reasoner simulators
  - **No**, because they **do not use logical rules** and they make **mistakes**
    - Focus on formal reasoners

# Is one better than the other one?

Depending on the use case scenario, one may want to employ a formal reasoner or a reasoner simulator

## Formal reasoner

- When we care about 100% correctness and consistency
- When we need a trace of the reasoning rules used to generate the output

## Reasoner simulator

- When we don’t care how the output is generated
- When we don’t care that the system is always consistent or correct

## Can they help each other?

**Reasoner simulators may be incorrect**

They can be combined with a reasoner for increased correctness, consistency, and reduced resource consumption

Ex. Plan-SOFAI, Rao’s approach of LLMs modulo

**Formal reasoners do not have a model of the world**

Unless we explicitly provide it to them

They can be combined with reasoner simulators to improve their flexibility and performance

Ex.: Alpha-geometry

# AI past and present

## First AI phase: High-level cognitive capabilities (reasoning)

- Logic, planning, search, optimization
- Pros: mimicking human high-level reasoning capabilities
- Cons: needs controlled working environments

## Second AI phase: Data-driven approaches (reasoner simulators)

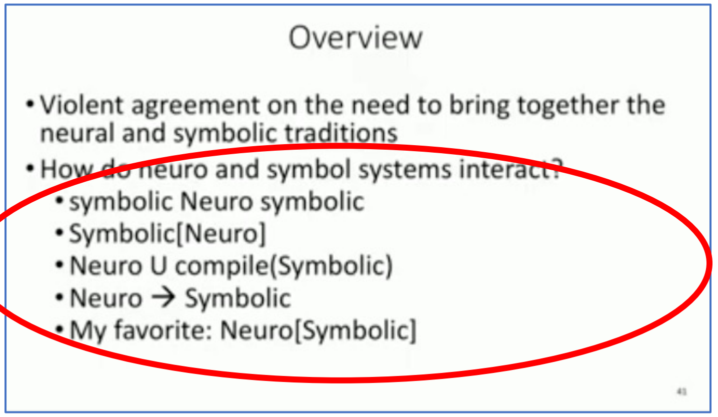
- Making sense of raw data to make predictions and classification
- Pros: perception, many applications, directly linked to real world
- Cons: lack of interpretability/explainability, data quantity and quality issues, hidden bias, lack of generalizability and robustness

## Current AI phase: Reunification

- Combine data-driven and symbolic logic-based approaches
- To make the next big advancement in AI capabilities
  - Both autonomous agents and decision-support systems
- Also to address some of the ethics issues

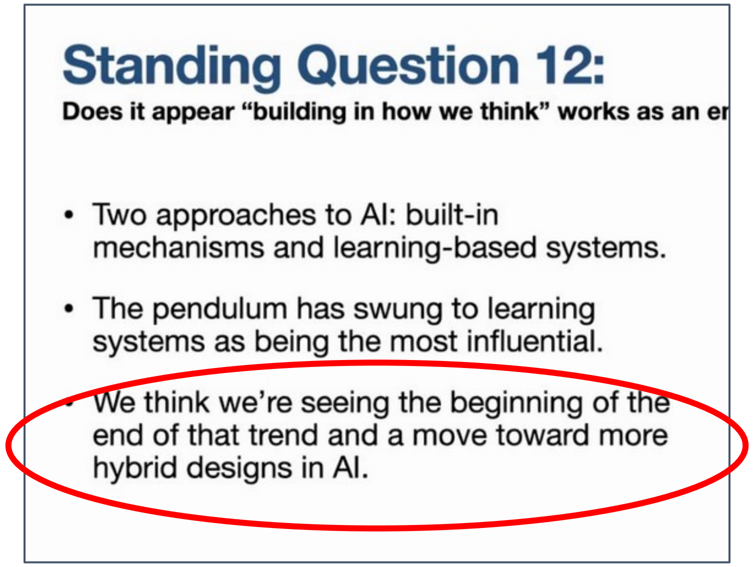
# Agreement on "what" but not on "how"

- The third AI summer, AAAI 2020 talk by Henry Kautz, Feb. 2020
- Neurosymbolic AI: The 3rd wave, L. Lamb and A. D'Avila Garcez, Dec. 2020
- Inductive biases for deep learning of higher-level cognition, A. Goyal and Y. Bengio, Feb. 2021
- AI 100 report, Sept. 2021
- Machine Learning and Logic: Fast and Slow Thinking, Moshe Vardi, 2022



Overview

- Violent agreement on the need to bring together the neural and symbolic traditions
- How do neuro and symbol systems interact?
  - symbolic Neuro symbolic
  - Symbolic[Neuro]
  - Neuro U compile(Symbolic)
  - Neuro → Symbolic
  - My favorite: Neuro[Symbolic]



**Standing Question 12:**

**Does it appear "building in how we think" works as an er**

- Two approaches to AI: built-in mechanisms and learning-based systems.
- The pendulum has swung to learning systems as being the most influential.
- We think we're seeing the beginning of the end of that trend and a move toward more hybrid designs in AI.

## Look (again) at the human mind and brain

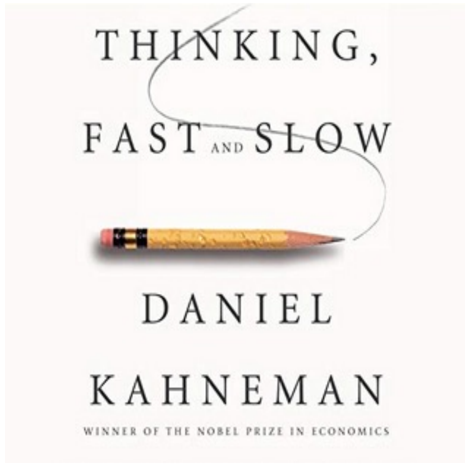
### Inspiration from neuroscience and cognitive science

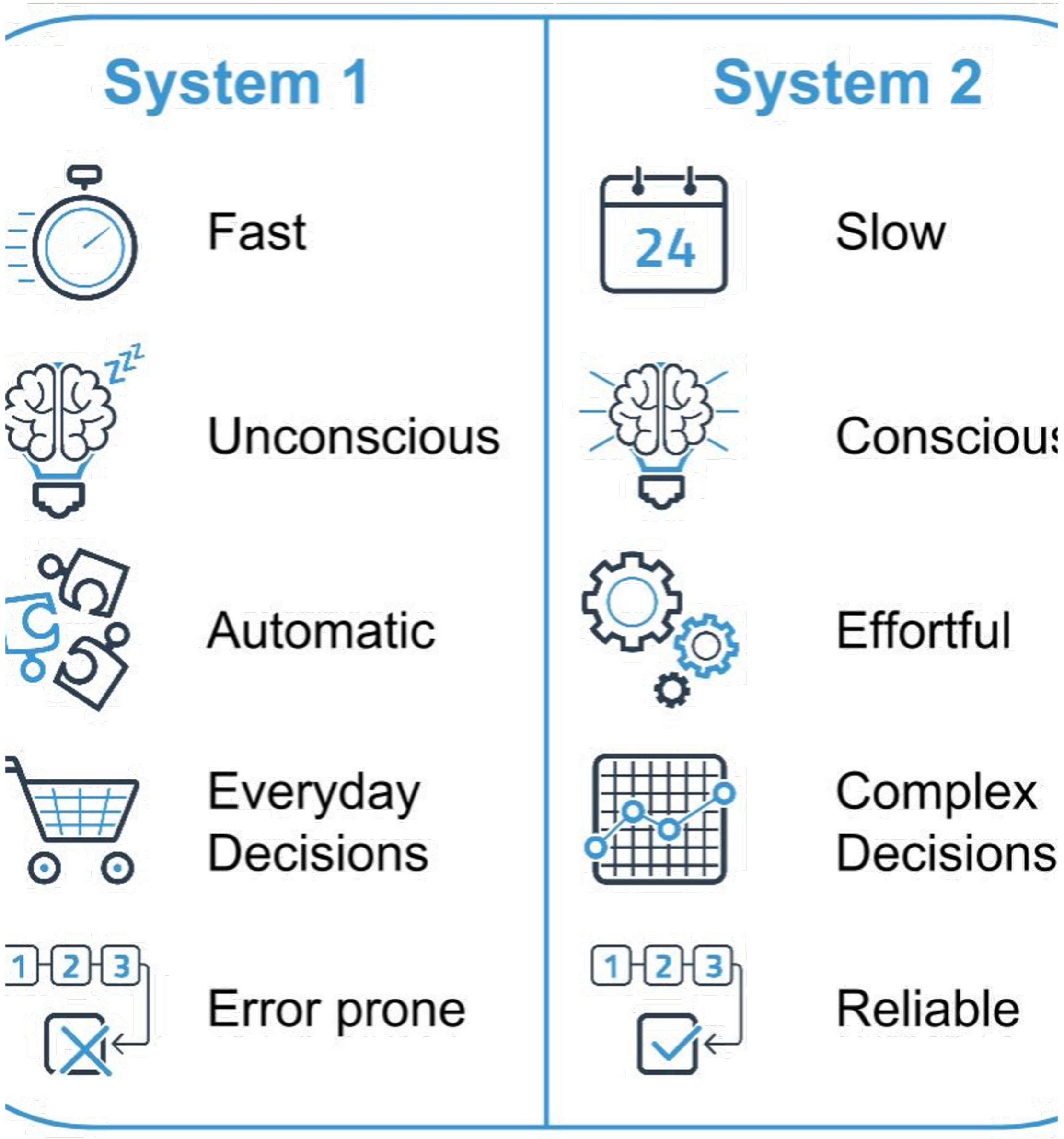
- Neuroscience can help identify effective AI structures for problem solvers
- Cognitive science can inspire AI architectures for solvers’ combination, arbitration, and governance

### Can it succeed now?

- Better knowledge of how mind and brain work
- More mature AI technology
  - Including more data, computing power, and applications

# Thinking Fast and Slow, or System 1 and System 2





| System 1 | System 2 |
|---|---|
| Fast | Slow |
| Unconscious | Conscious |
| Automatic | Effortful |
| Everyday Decisions | Complex Decisions |
| Error prone | Reliable |

# Thinking Fast and Slow, or System 1 and System 2

System 1 can perform cognitively easy tasks

- Local and parallel
- Handles and exploits causality to build approximate models of the world

System 2 handles more complex tasks

- Requires all our attention
- Global and sequential

System 2's search for solutions is usually supported by System 1's heuristics

Skill learning: some tasks transfer from System 2 to System 1 over time

- Not all (ex.: complex arithmetic operations)

Cognitive control

- System 1 reacts, System 2 may override it

## Daniel Kahneman on current AI (Dec. 2023)

“Since the triumphs of machine learning in 2012, artificial intelligence produces solutions by the opaque operations of large neural net, which shares many features with the automatic intuitions of System 1. The integration of intuition and reasoning in artificial intelligence was not achieved during the first decade of the machine-learning era."

# Thinking fast and slow in AI: our approach

## A multi-agent architecture -- SOFAI

### System 1 solvers

- Relying on past experience and model of the world
- Not reasoning on the problem
- Reacting to arrival of new problem instance
- Generating a candidate solution

### System 2 solvers

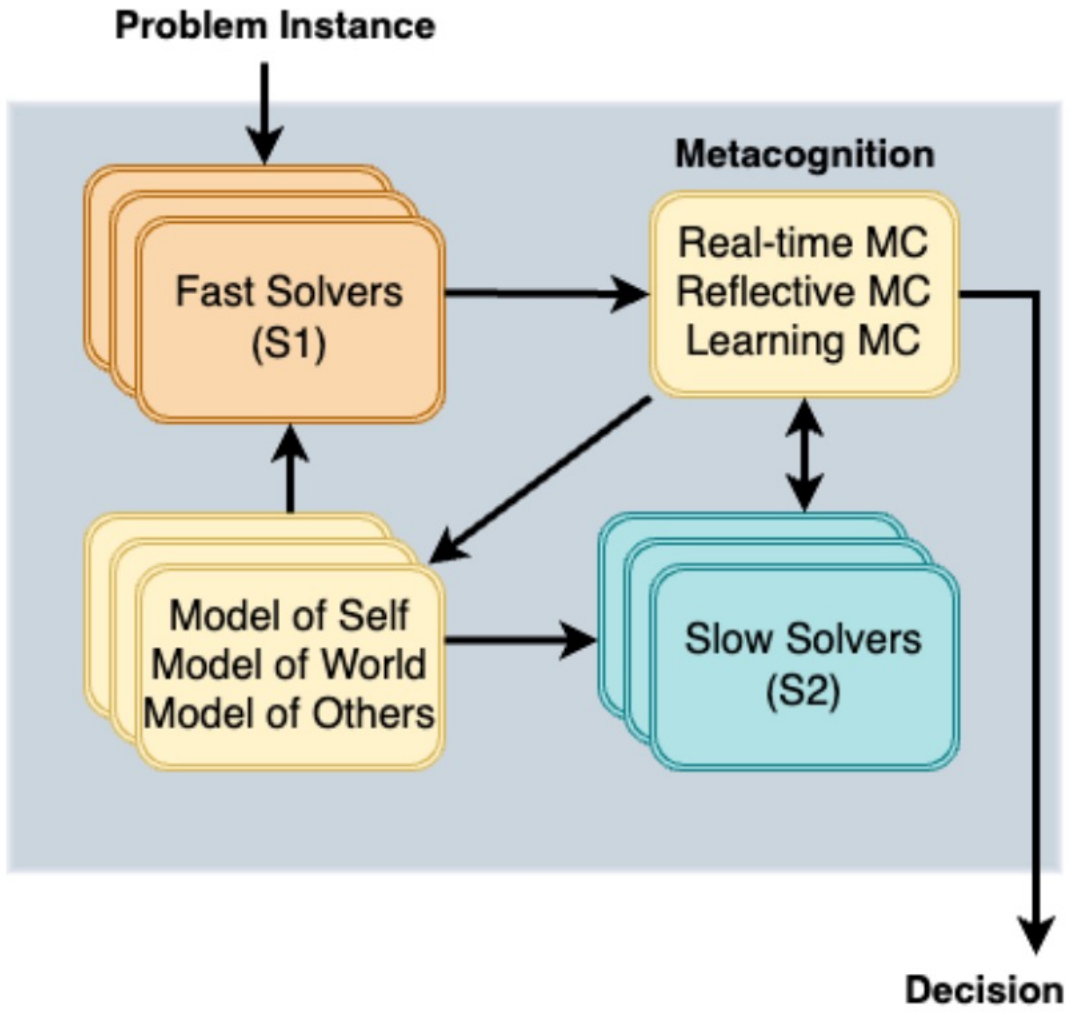
- Reasoning on the given problem
- Computational complexity dependent on the size of input
- Activated by meta-cognition
- Generating solution

### Model/solver updater

- Acts in background to update models of world, self, and others

# The (basic) SOFAI architecture

- **Fast solvers**: only based on **past experience**
- **Slow solvers**: **reasoning** about the problem instance
- **Metacognition**:
  - **Real-time**: decides which solver to use
  - **Offline** – **reflective**: counterfactual S2 comparison and possible adjustment of MC parameters
  - **Offline** – **learning**: solvers/models update
- Fast solvers act independently, slow solvers need to be triggered by MC
- Note: Inspiration from theories of the mind, rather than the brain



## Two phases for Real-time MC

Input: System 1 solution and confidence

Output: adopt System 1 solution or activate System 2

Goal: avoid unnecessary costly reasoning processes unless the additional expected reward gain is higher than the expected higher cost

MC1

- Enough resources for MC1 and MC2? If not, adopt S1’s solution
- Enough experience accumulated? If yes, adopt S1’s solution
- S1 confidence higher than expected reward? If yes, adopt S1’s solution

MC2

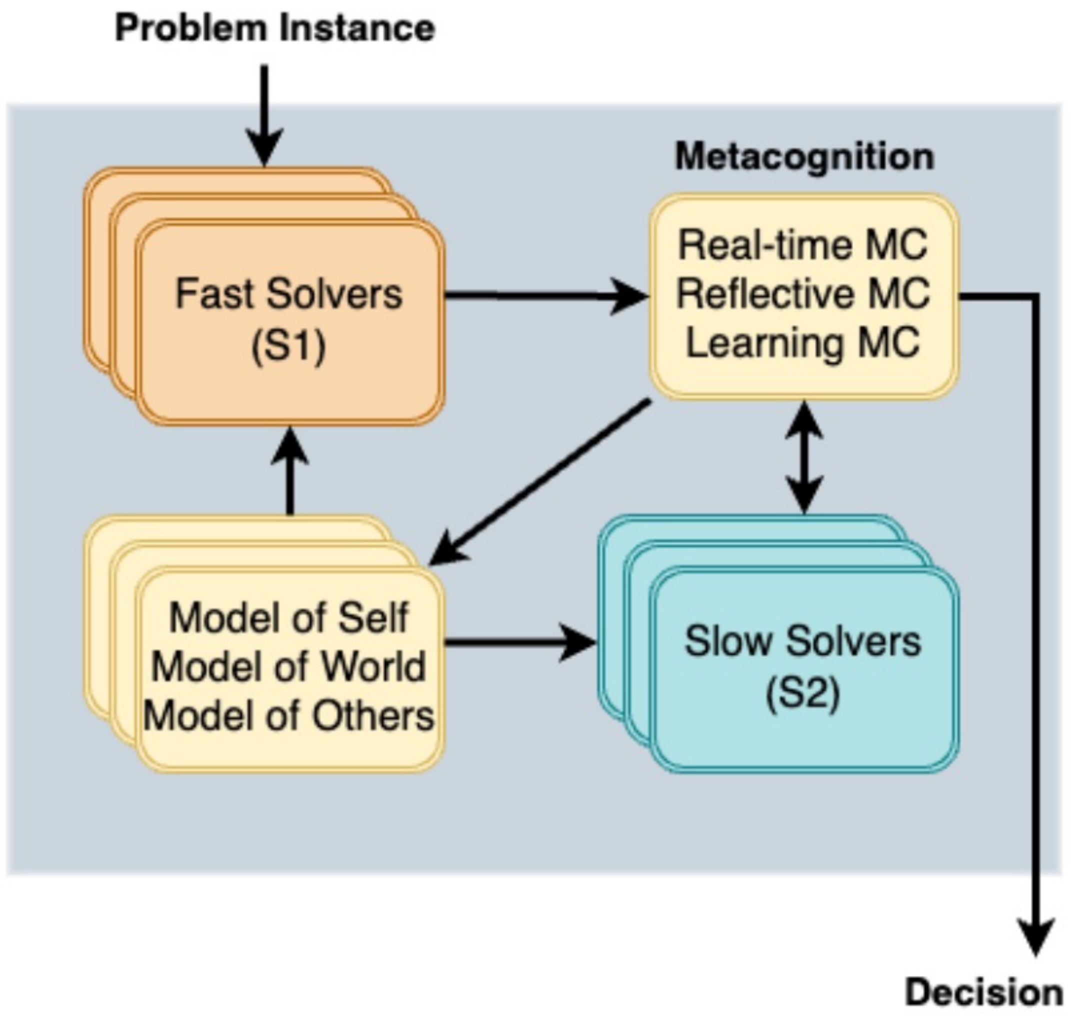
- Exploration: with small probability, choose randomly between S1’s solution and S2’s activation
- Expected reward gain for S2 solver > expected higher cost of using S2 ➔ activate S2, otherwise adopt S1’s solution

## SOFAI vs Main Neuro-symbolic Approaches

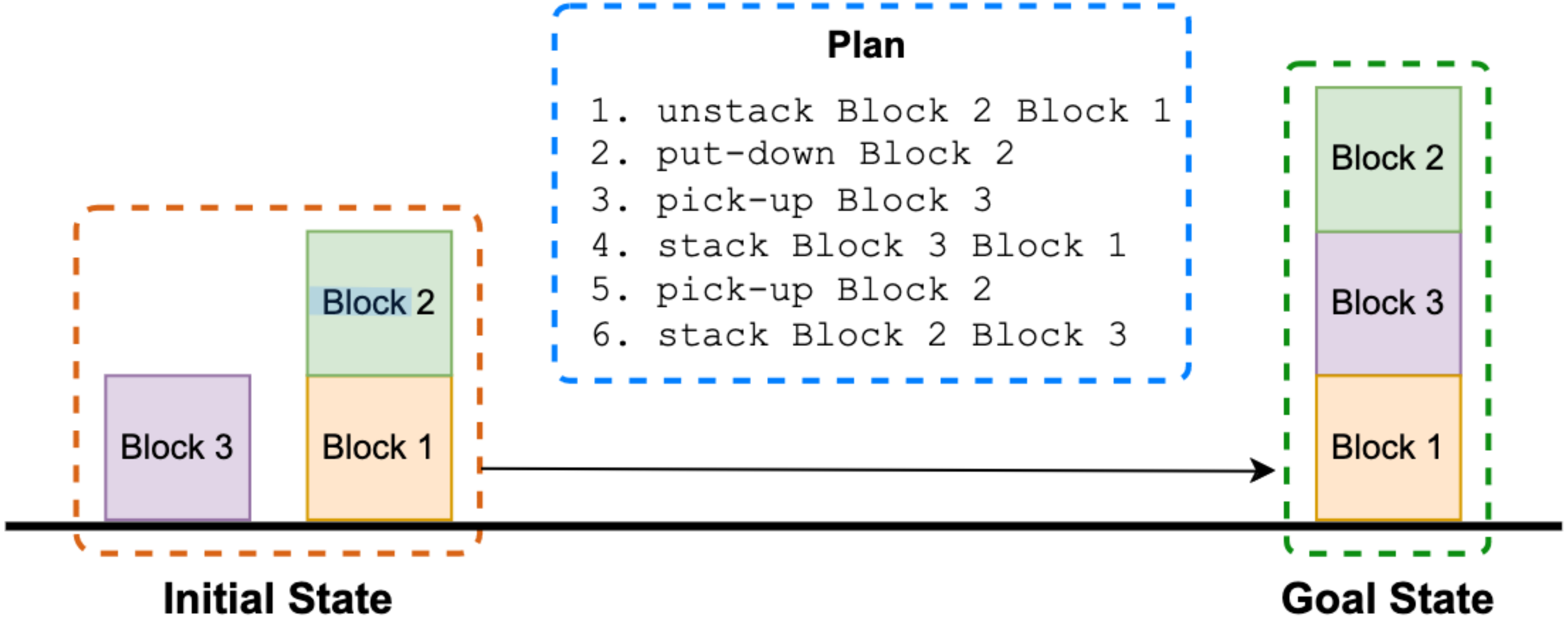
- Full integration
  - Logical Neural Nets, Tensor Neural Nets, …
- Combination of neuro- and symbolic components, decided by the researchers
  - Ad hoc for each specific task
- SOFAI adopts the second approach but the metacognition component provides the governance

# Expected SOFAI Behavior

- Make **superior decisions** than each of the solvers independently
- **Improve over time**
- **Internalize skills** (from S2 to S1)
- Exercise **cognitive control** in **high-risk** scenarios
- Leverage **existing solvers**
- **Adapt** to the solvers' competence/capabilities



# Symbolic planning

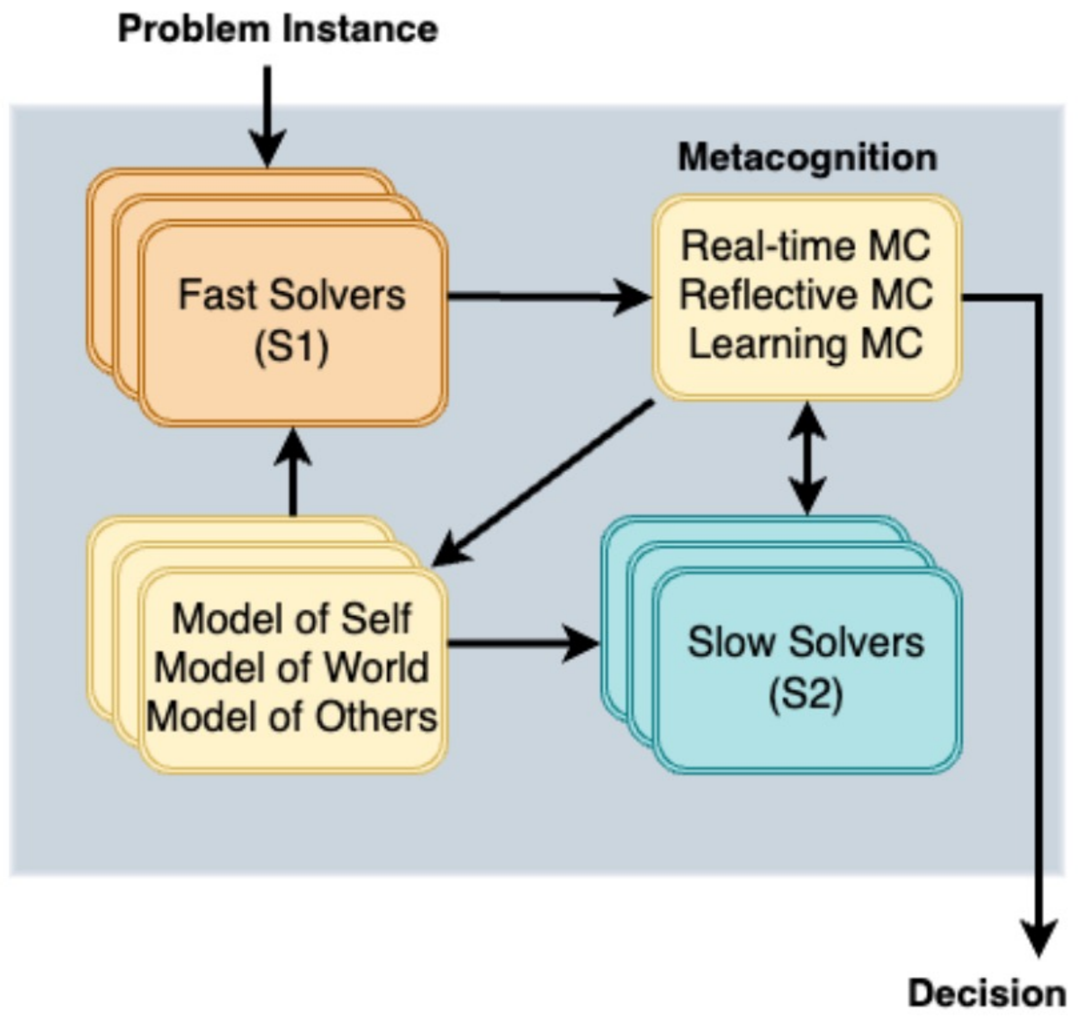


**Plan**

```
1. unstack Block 2 Block 1
2. put-down Block 2
3. pick-up Block 3
4. stack Block 3 Block 1
5. pick-up Block 2
6. stack Block 2 Block 3
```

# SOFAI for planning

- S1 solver: an LLM-based planner (Plansformer)
- S2 solvers: existing classical symbolic planners, as **FD** (Fast Downwards) and **LPG** (Local Search for Planning Graphs with Action Costs)
- Metacognition:
  - Only the **real-time version**
  - Includes **plan validation**



# Capabilities of LLMs for Plan Generation – experimental results

- To what extent can different LLM architectures solve planning problems?

  `Encoder-decoder` and `decoder-only` models show better plan generation

- What pre-training data is effective for plan generation?

  LLMs pre-trained on programming code are better at plan generation

- Does fine-tuning and prompting improve plan generation?

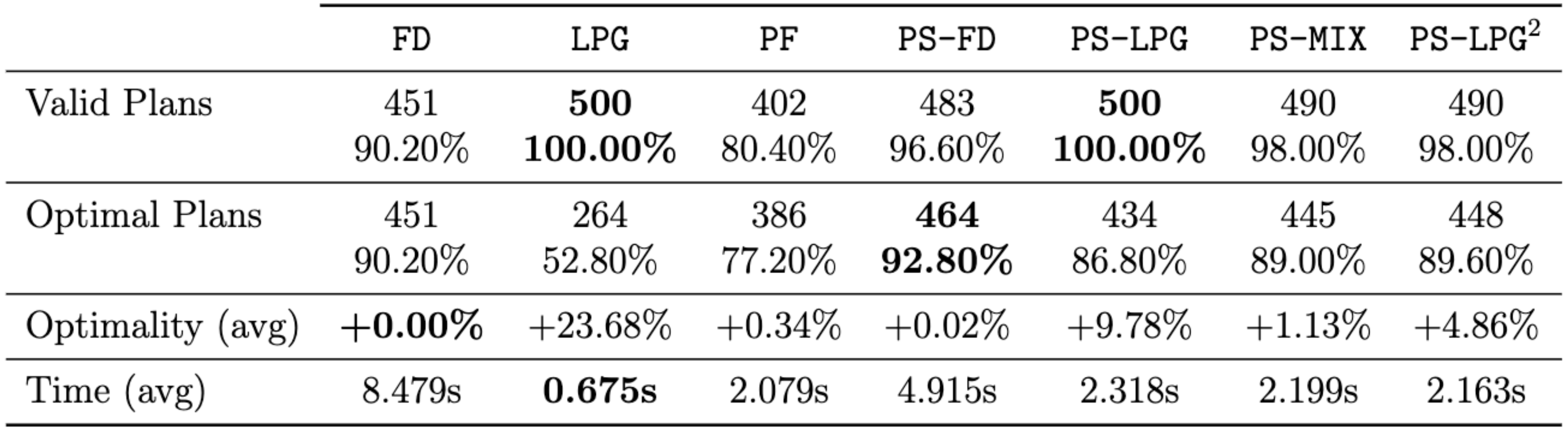
  Fine-tuning is better than prompting

- Are LLMs capable of generalization?

  Incapable of generalization (length, object names, new domains)

Pallagani et al. "On the prospects of incorporating Large Language Models (LLMs) in automated planning and scheduling (aps)." *ICAPS* (2024)

# Plan-SOFAI Experimental Results

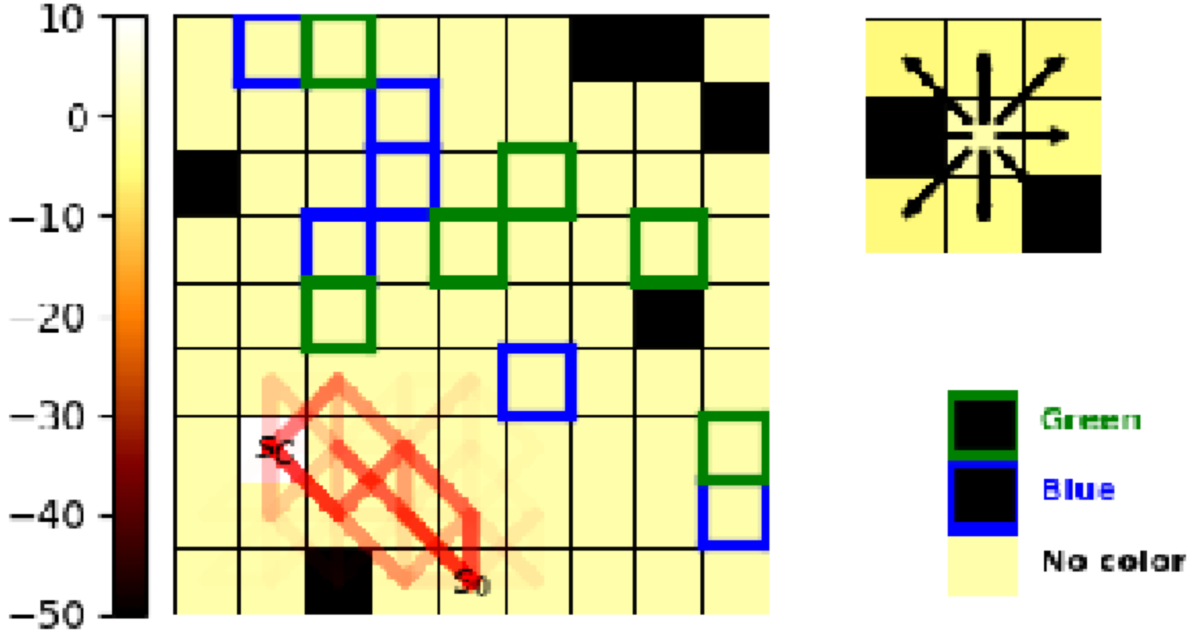


| | FD | LPG | PF | PS-FD | PS-LPG | PS-MIX | PS-LPG$^2$ |
|---|---|---|---|---|---|---|---|
| Valid Plans | 451<br>90.20% | **500**<br>**100.00%** | 402<br>80.40% | 483<br>96.60% | **500**<br>**100.00%** | 490<br>98.00% | 490<br>98.00% |
| Optimal Plans | 451<br>90.20% | 264<br>52.80% | 386<br>77.20% | **464**<br>**92.80%** | 434<br>86.80% | 445<br>89.00% | 448<br>89.60% |
| Optimality (avg) | **+0.00%** | +23.68% | +0.34% | +0.02% | +9.78% | +1.13% | +4.86% |
| Time (avg) | 8.479s | **0.675s** | 2.079s | 4.915s | 2.318s | 2.199s | 2.163s |

- **FD** has the **best optimality** (+0.00%) but **high** overall **resource** consumption (8.479 sec)
- **LPG** is the **most efficient** (0.675 sec) but has **less**-than-**ideal optimality** statistics (+23.68%)
- **PF** solves less problems (402) than FD and LPG but many (386) are optimal and time is constant (2.079 sec)
- Among the SOFAI instances:
  - **PS-MIX** represents the most **balanced trade-off** among all the analyzed techniques
    - 490 solved problems, 2.199 sec, +1.13% optimality
  - **PS-LPG** best if we care less about optimality (all problems solved, 434 optimal)

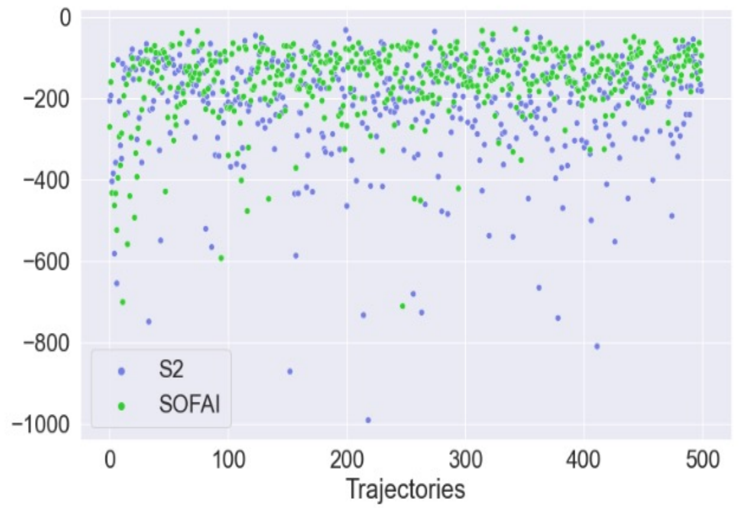
# SOFAI for Constrained Grid Navigation

- **Constraints** over **cells** and **moves** (actions)
- **Goals**:
  - Maximize reward, minimize time and length
  - High risk aversion: minimize constraint violation
- **Solvers**:
  - S1 solver: RL agent, trained on past trajectories and their reward
  - S2 solver: MDFT (deliberate decision-making)
- **Meta-cognition:**
  - Real-time MC: two-phases choice between S1 and S2
  - Reflective (counterfactual) MC: compare past trajectories with simulated S2-only ones, and adjust MC parameters
  - Learning MC: update model of self and S1 solver
- Five versions of S1 solver, with varying degrees of randomness

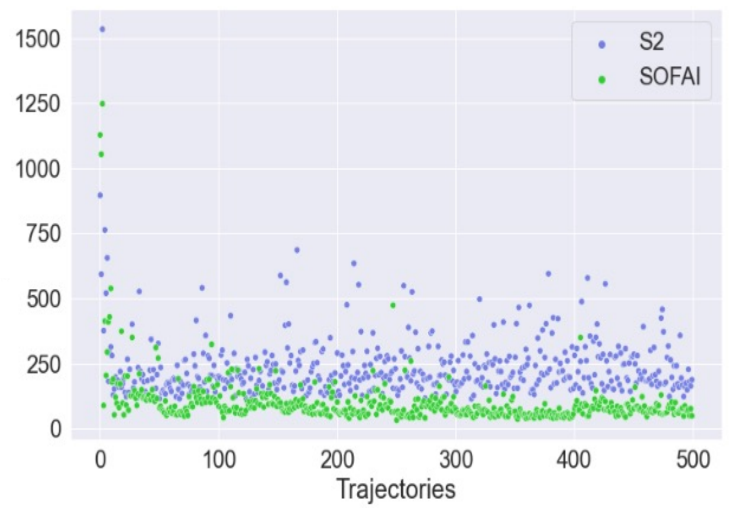


# SOFAI is better than using S1 or S2 independently

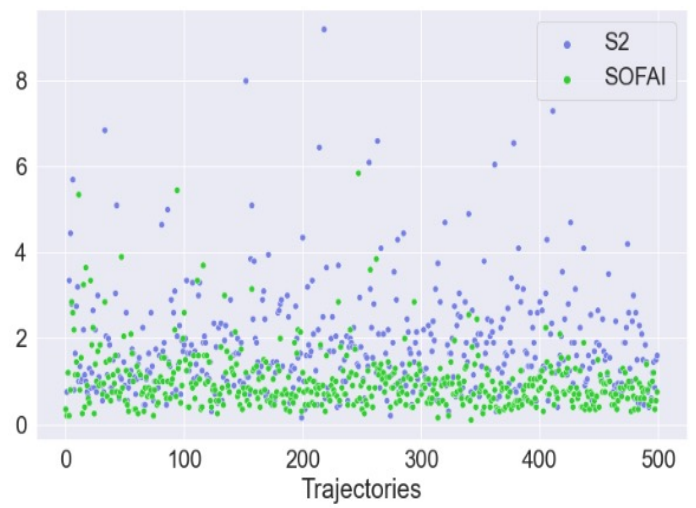
Reward



Time

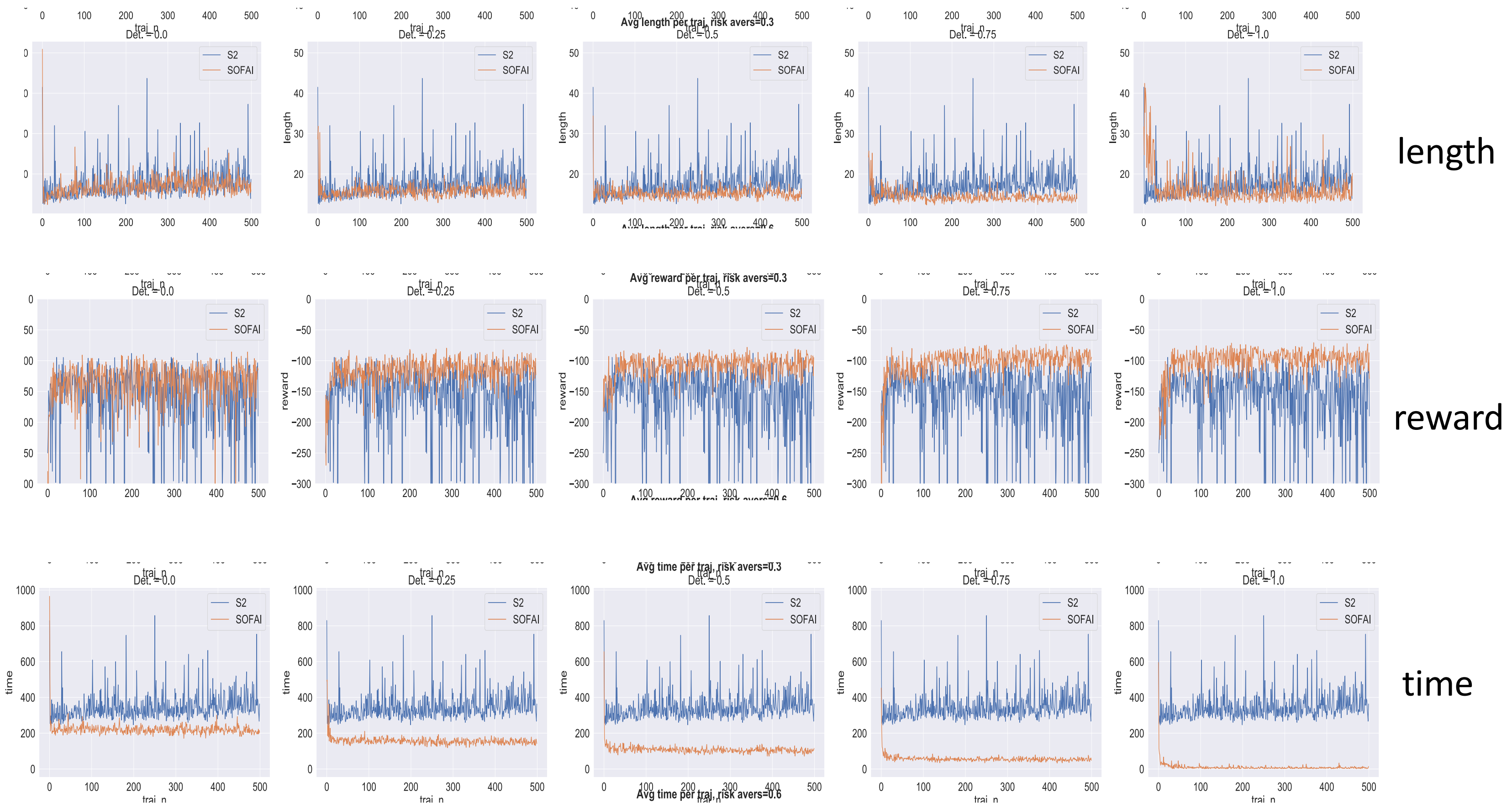


Length

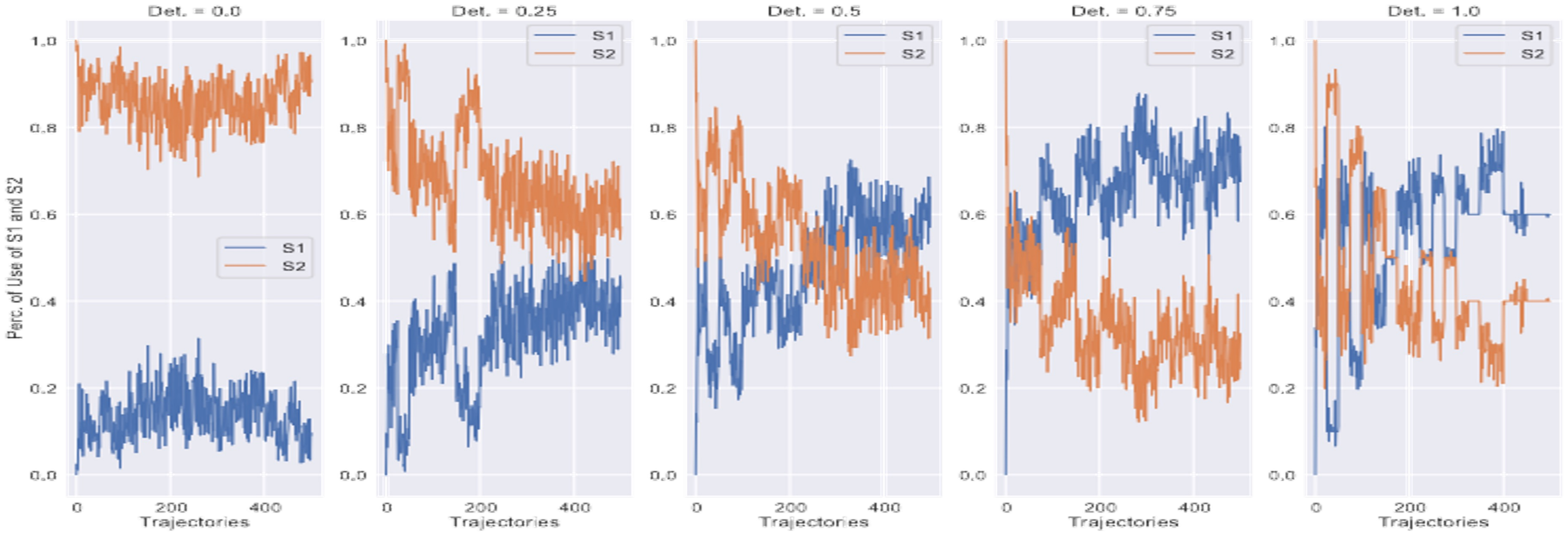


# How: SOFAI adapts to the competence of the S1 solver

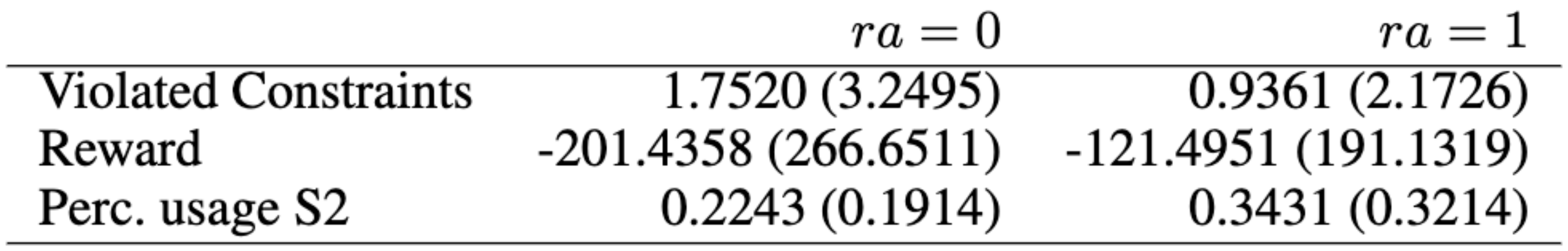
- **SOFAI adapts to the capabilities of the S1 solver**
  - Five versions of S1 solver, from completely random to always choosing move with best expected reward
  - Reward is (almost) independent
  - S1 is used less if SOFAI realizes that it is not reliable
- **Over time, SOFAI learns to calibrate the use of S1 and S2 to maintain a stable trajectory quality**



# How: SOFAI’s skill learning (from mostly S2 to mostly S1)



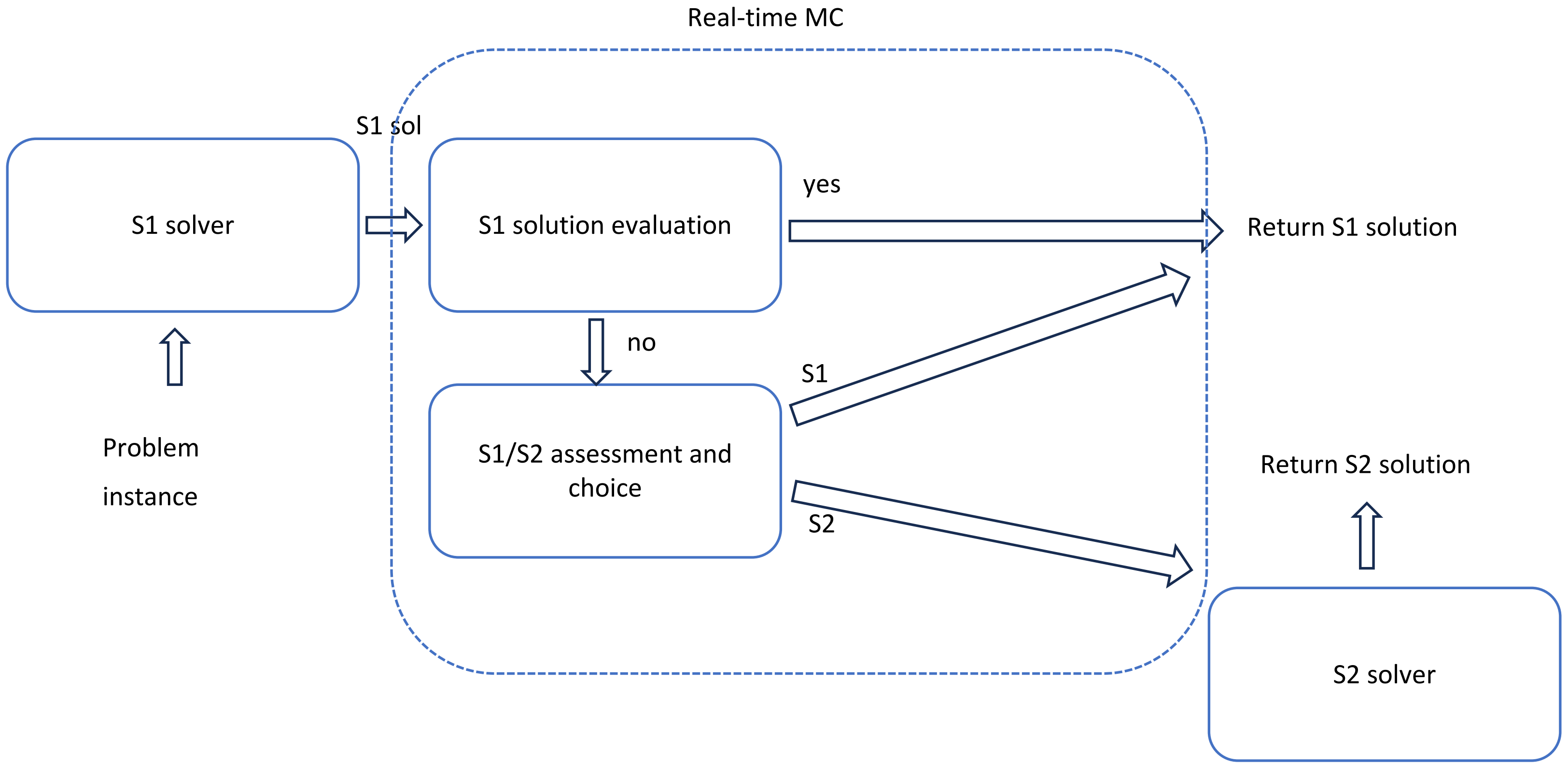
## How: SOFAI cognitive control (more cautious in the face of high risk aversion)



| | $ra = 0$ | $ra = 1$ |
|---|---|---|
| Violated Constraints | 1.7520 (3.2495) | 0.9361 (2.1726) |
| Reward | -201.4358 (266.6511) | -121.4951 (191.1319) |
| Perc. usage S2 | 0.2243 (0.1914) | 0.3431 (0.3214) |

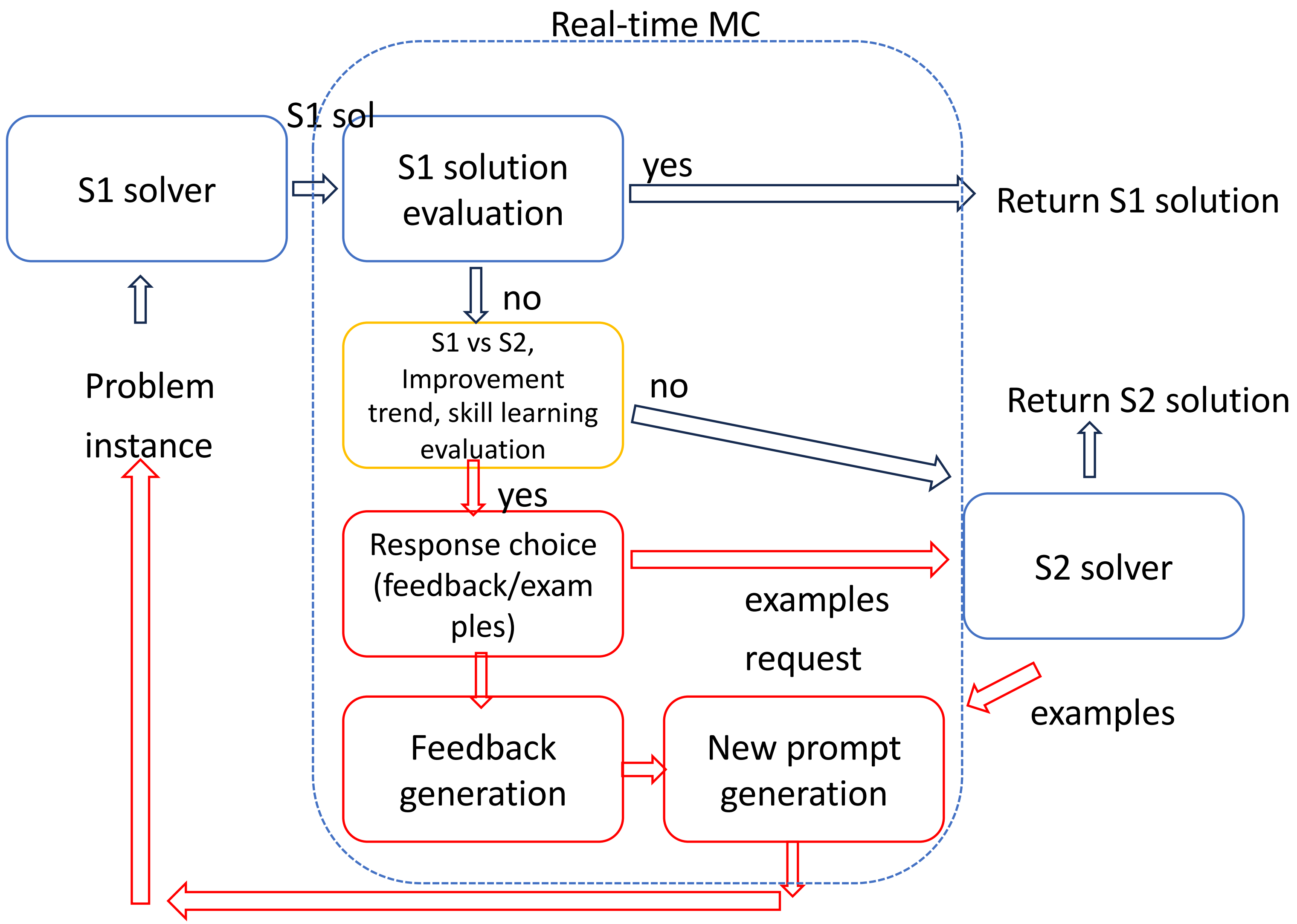
# A new look at the basic SOFAI architecture

- MC evaluates the S1 candidate solution: if not acceptable, calls the S2 solver



# The generalized SOFAI architecture

- Main change: use the S1 solver more than once, before calling the S2 solver
- S1 solver: LLM
- MC to provide feedback on LLM's candidate solution
- Leverage S2 solver to provide examples to the LLM
    - Not just to solve the entire problem instance



# Generalized SOFAI for graph coloring

- **S1 solver**: an LLM (Mistral 7B )
- **S2 solver**: DSatur Algorithm
  - Not an optimization algorithm, may use more colors than needed
- **Real-time metacognition**
  - Validates the color assignment generated by the LLM
  - Provides feedback: Right/Wrong, Single Mistake, All Mistakes
- Example types: None, Pair, Triangle, Pair and Triangle, 4-Clique
- At most 5 calls to the S1 solver before calling the S2 solver

- Experimental setting:
  - Graph sizes : 5-9 nodes
    - Edges generated randomly to connect all nodes and add some additional edges
  - 100 graphs per experiment
    - 75 different experiments, combining 3 feedback types, 5 example types, and 5 graph sizes

## LLM prompt

Note: upper bound on number of colors given by the S2 solver.

Graph Coloring Problem: You are provided with an undirected graph. The graph is represented by vertices and edges connecting them. In the examples below, vertices are labeled with letters (e.g., A, B, C), and edges are represented by pairs of vertices (e.g., A - B means there is an edge between vertex A and vertex B). Use the actual graph provided below for solving the problem:

R - Z

X - S

Y - R

Z - X

You can use at most 2 colors to generate a valid coloring of this graph such that no two adjacent vertices share the same color. ONLY OUTPUT the color assignments for each vertex in the following format: (vertex color). Example format of the output (Note: A, B, C are just examples): (A 1) (B 2) (C 1)

Use the vertex names from the actual graph content above when generating your solution.

# LLM and Validator Output

**Response from LLM/ Input to Validator:**

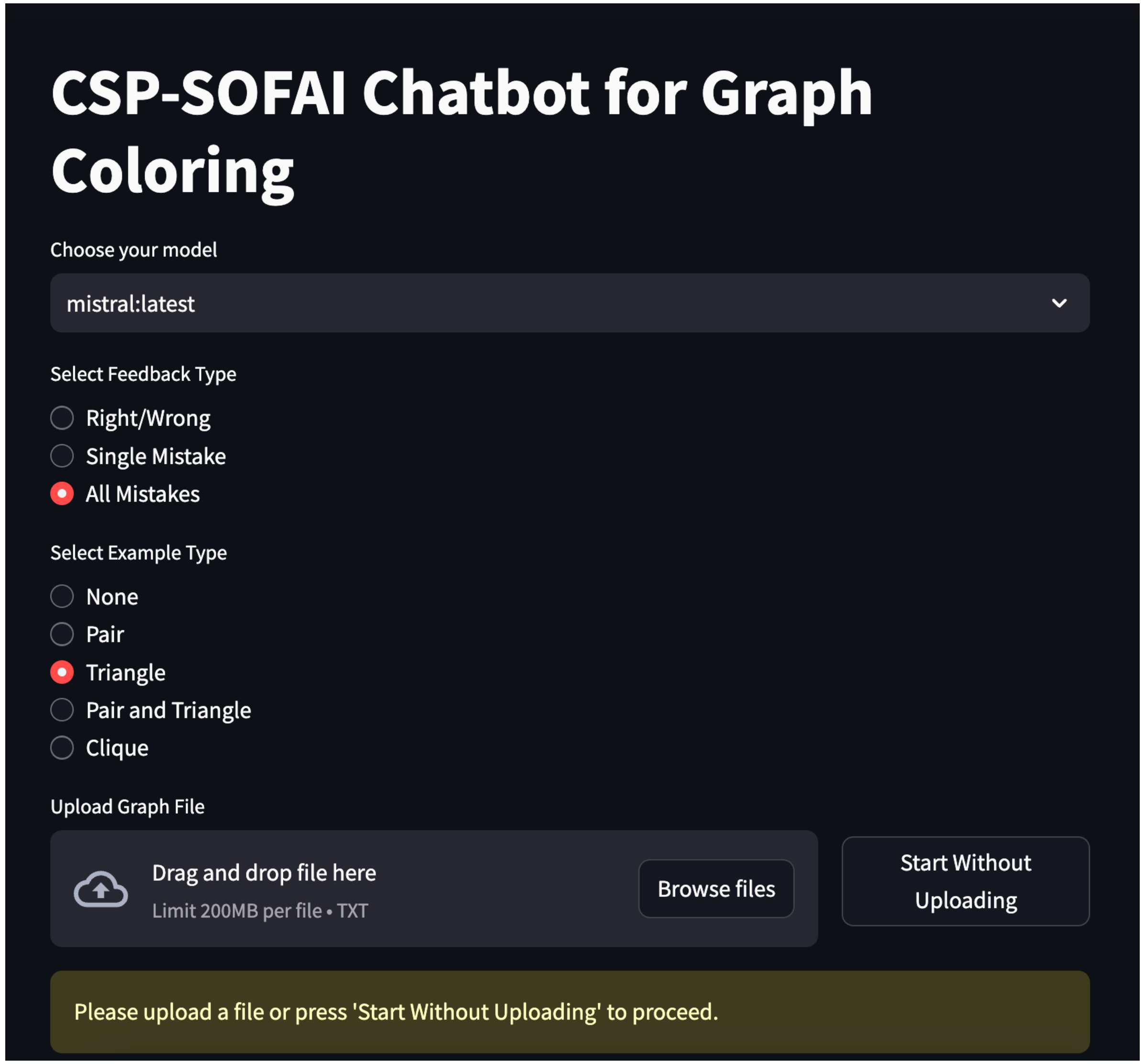
(R 1) (Z 2) (X 1) (S 2) (Y 1)

**Validator Output/ Return prompt:** (Feedback type single mistake and example type triangle)

The coloring is not correct. Feedback: Vertices R and Y are adjacent and share the same color 1.

For example, consider a graph coloring problem for three nodes, A, B, and C, all connected to each other (forming a fully connected subgraph, or clique). To color these nodes, assign different colors to each one. For instance, A can be colored with color 1, B with color 2, and C with color 3. This ensures that no two adjacent nodes share the same color. Example: (A 1) (B 2) (C 3)

# CSP-SOFAI demo screenshot



## CSP-SOFAI Chatbot for Graph Coloring

Choose your model

mistral:latest

Select Feedback Type

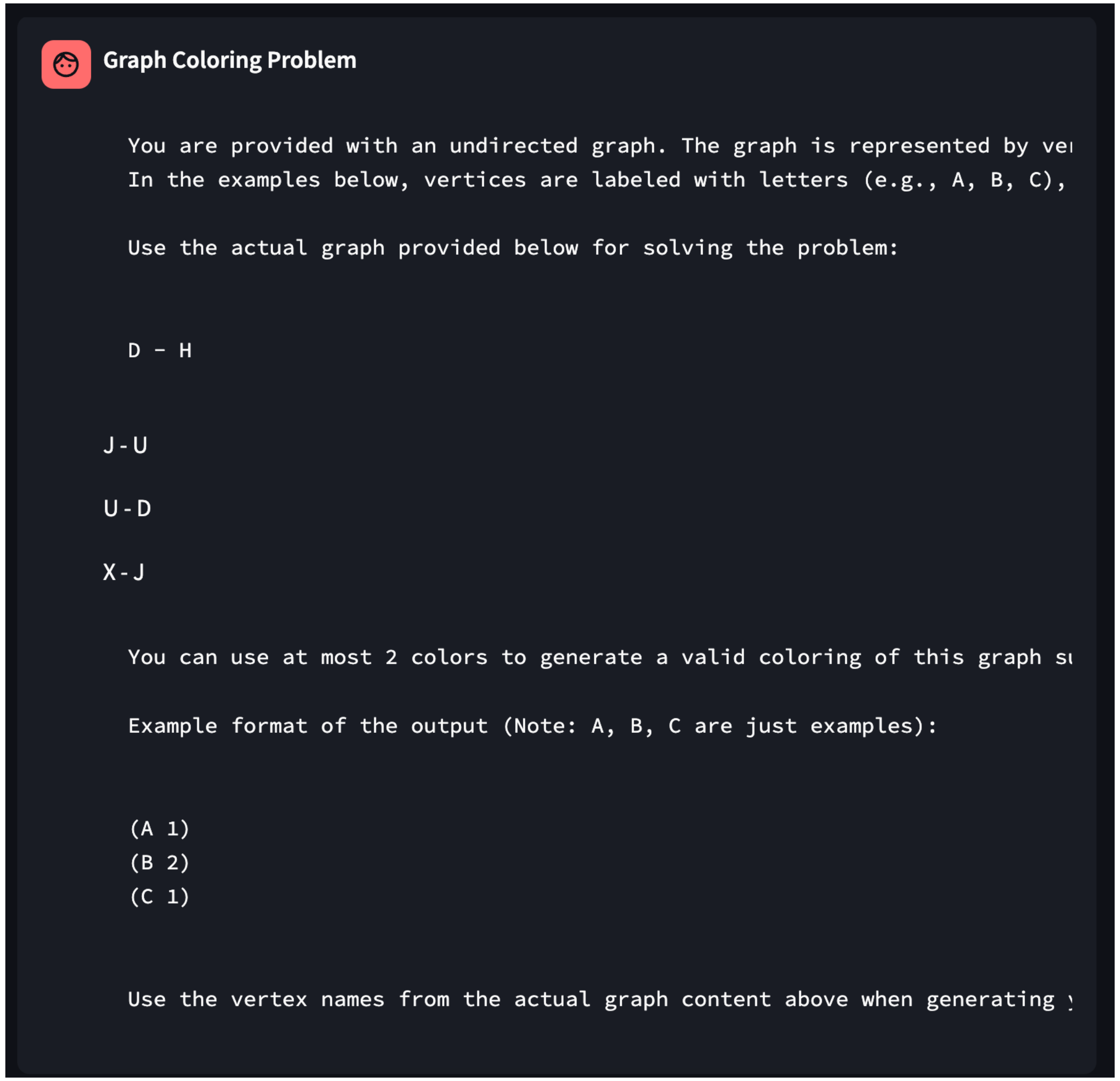
- ( ) Right/Wrong
- ( ) Single Mistake
- [✓ All Mistakes]

Select Example Type

- ( ) None
- ( ) Pair
- [✓ Triangle]
- ( ) Pair and Triangle
- ( ) Clique

Upload Graph File

Please upload a file or press 'Start Without Uploading' to proceed.



**Graph Coloring Problem**

```
You are provided with an undirected graph. The graph is represented by ve
In the examples below, vertices are labeled with letters (e.g., A, B, C),

Use the actual graph provided below for solving the problem:


D - H
```

J - U

U - D

X - J

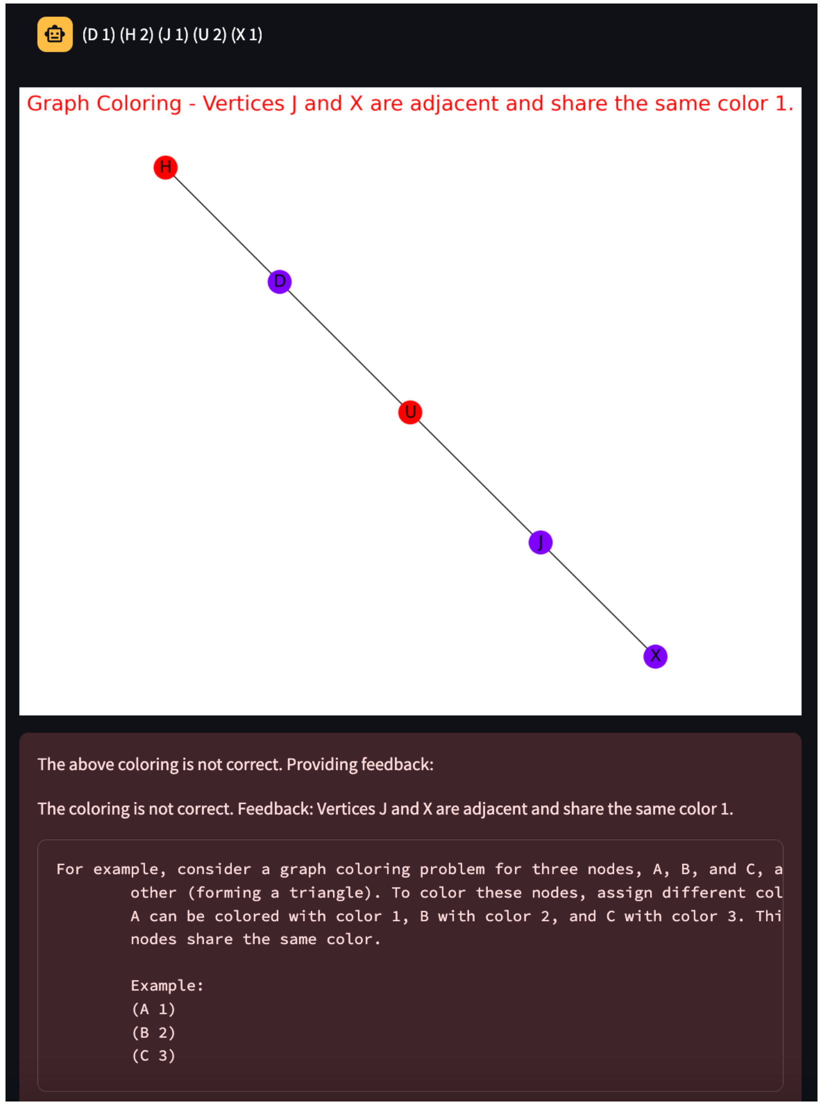
```
You can use at most 2 colors to generate a valid coloring of this graph s

Example format of the output (Note: A, B, C are just examples):


(A 1)
(B 2)
(C 1)


Use the vertex names from the actual graph content above when generating
```

# CSP-SOFAI demo Screenshot



(D 1) (H 2) (J 1) (U 2) (X 1)

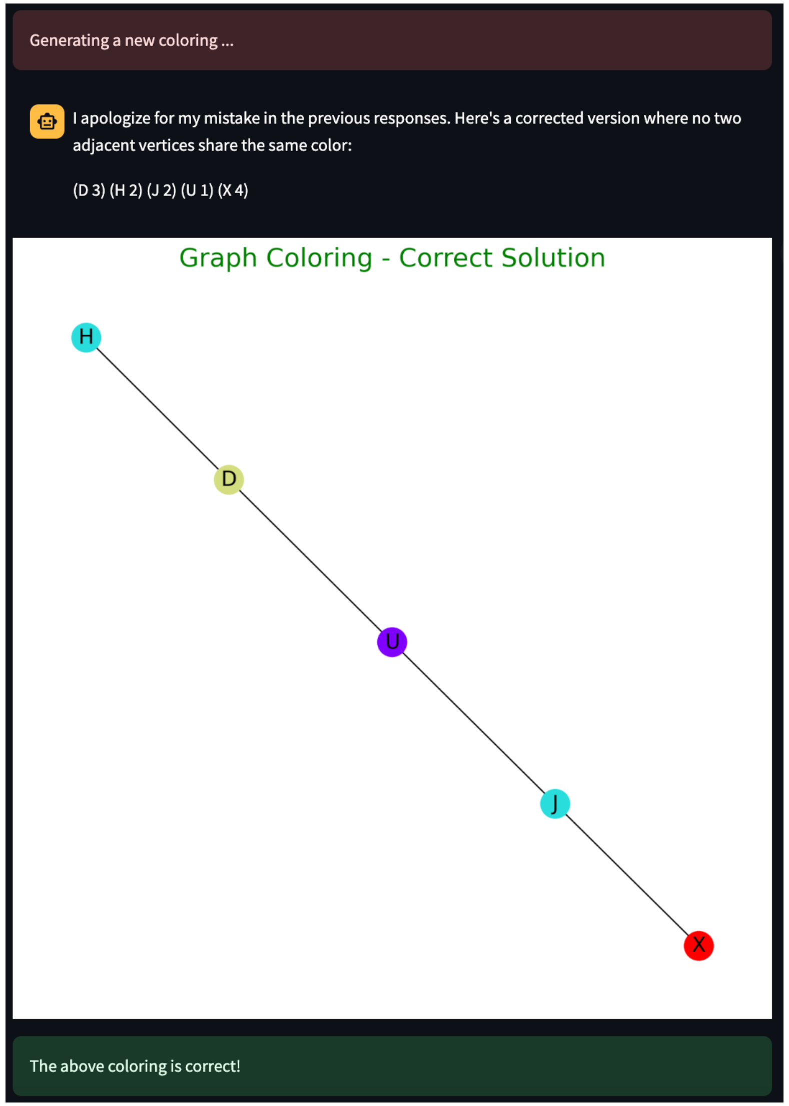
Graph Coloring - Vertices J and X are adjacent and share the same color 1.

The above coloring is not correct. Providing feedback:

The coloring is not correct. Feedback: Vertices J and X are adjacent and share the same color 1.

```
For example, consider a graph coloring problem for three nodes, A, B, and C, a
        other (forming a triangle). To color these nodes, assign different col
        A can be colored with color 1, B with color 2, and C with color 3. Thi
        nodes share the same color.

        Example:
        (A 1)
        (B 2)
        (C 3)
```



Generating a new coloring ...

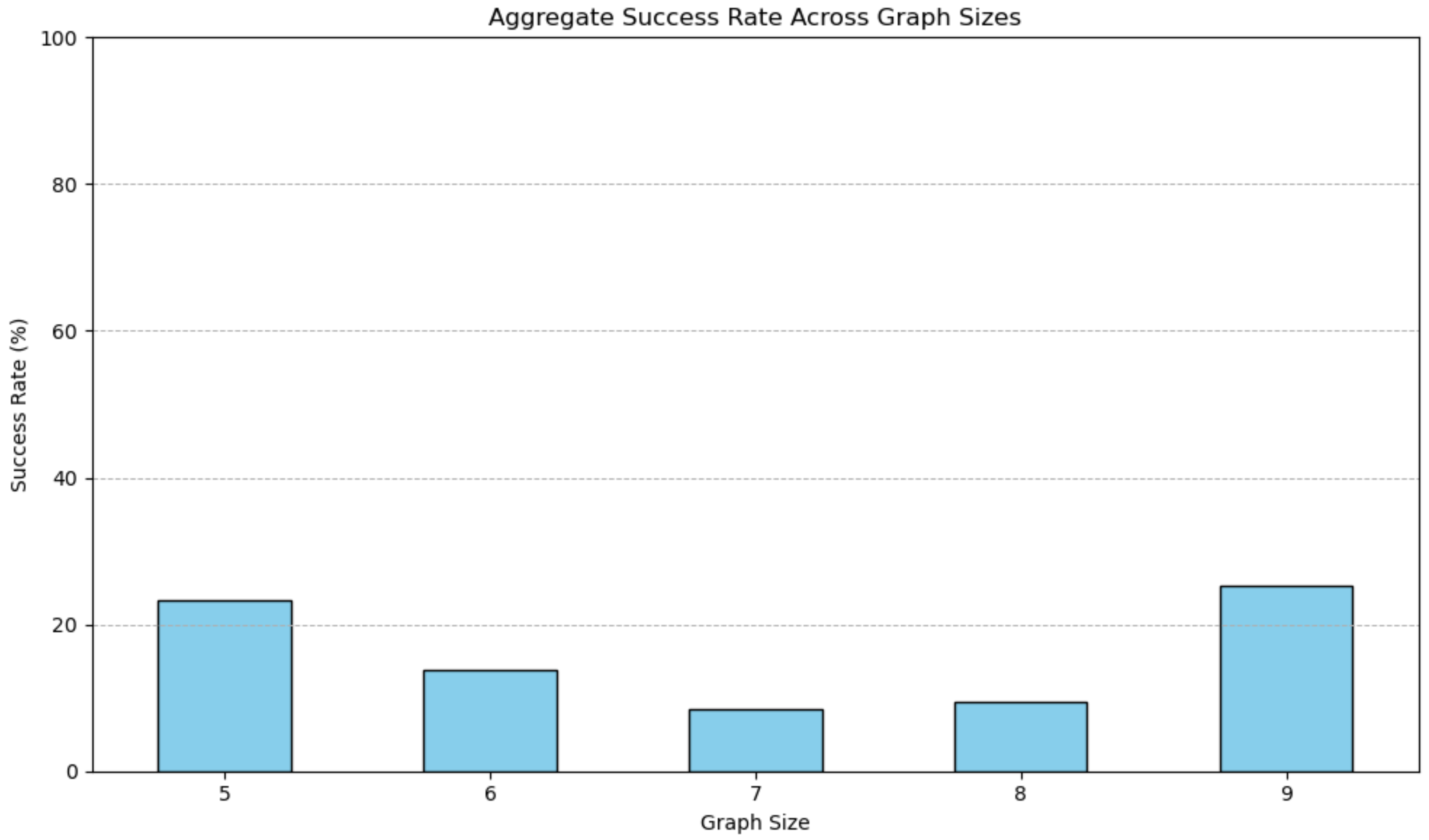
I apologize for my mistake in the previous responses. Here's a corrected version where no two adjacent vertices share the same color:

(D 3) (H 2) (J 2) (U 1) (X 4)

Graph Coloring - Correct Solution

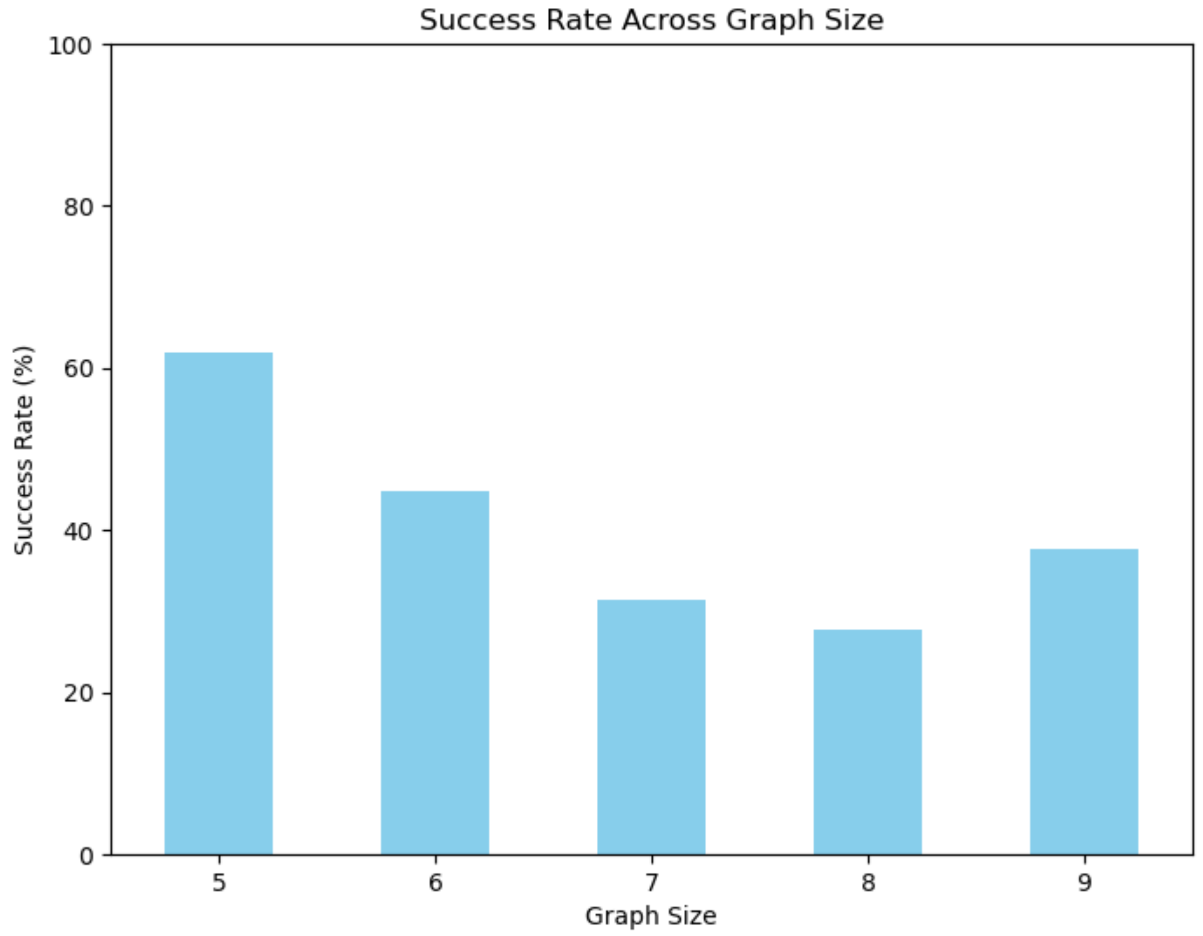
The above coloring is correct!

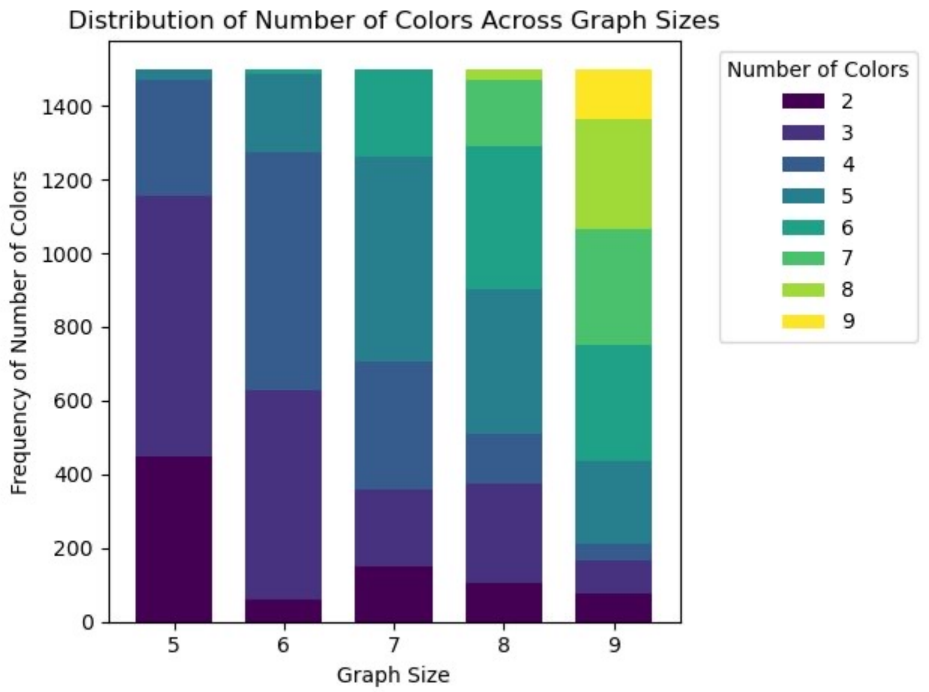
# Baseline – System 1 Only



The LLM (S1 solver)in one shot can solve at most 25% of the coloring problems.

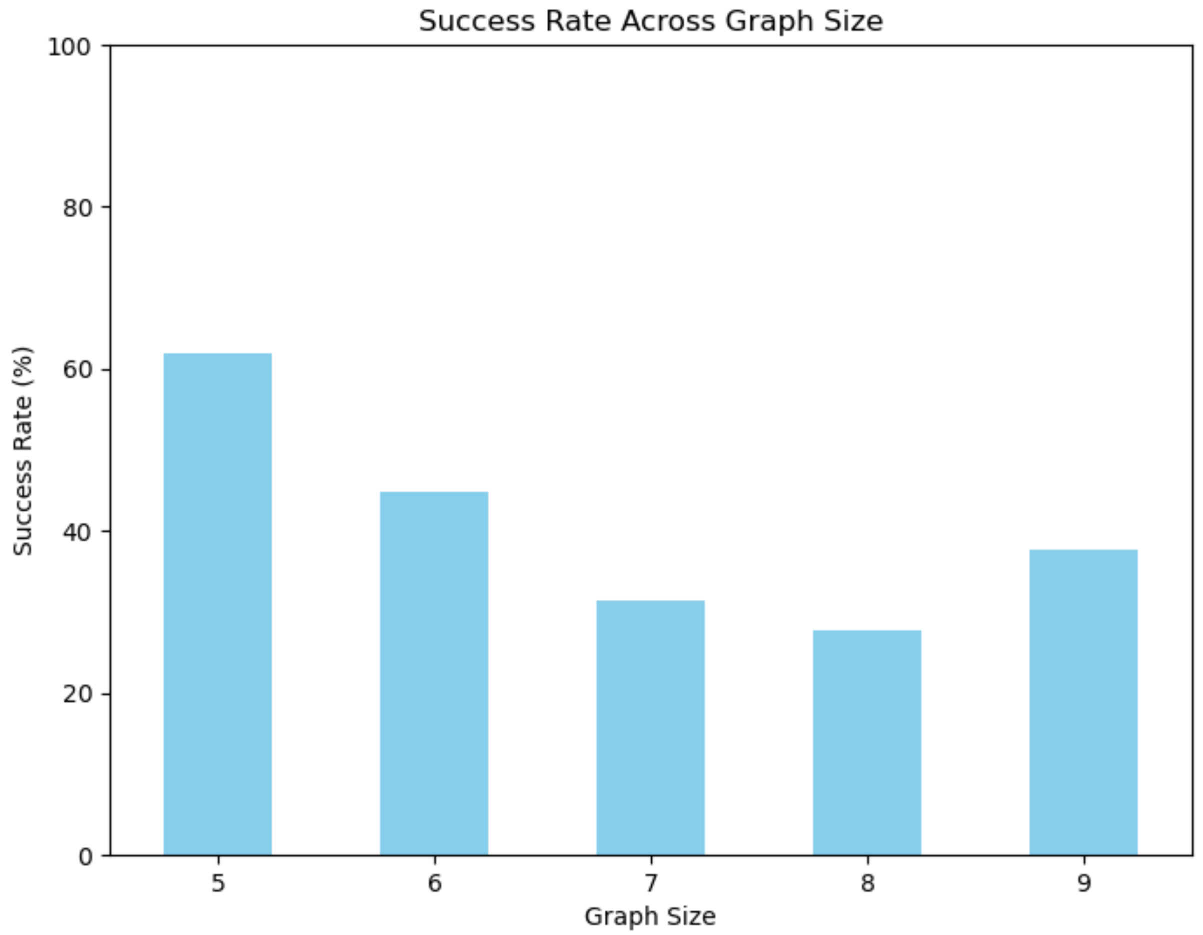
# SOFAI Success Rate – Across Graph Sizes

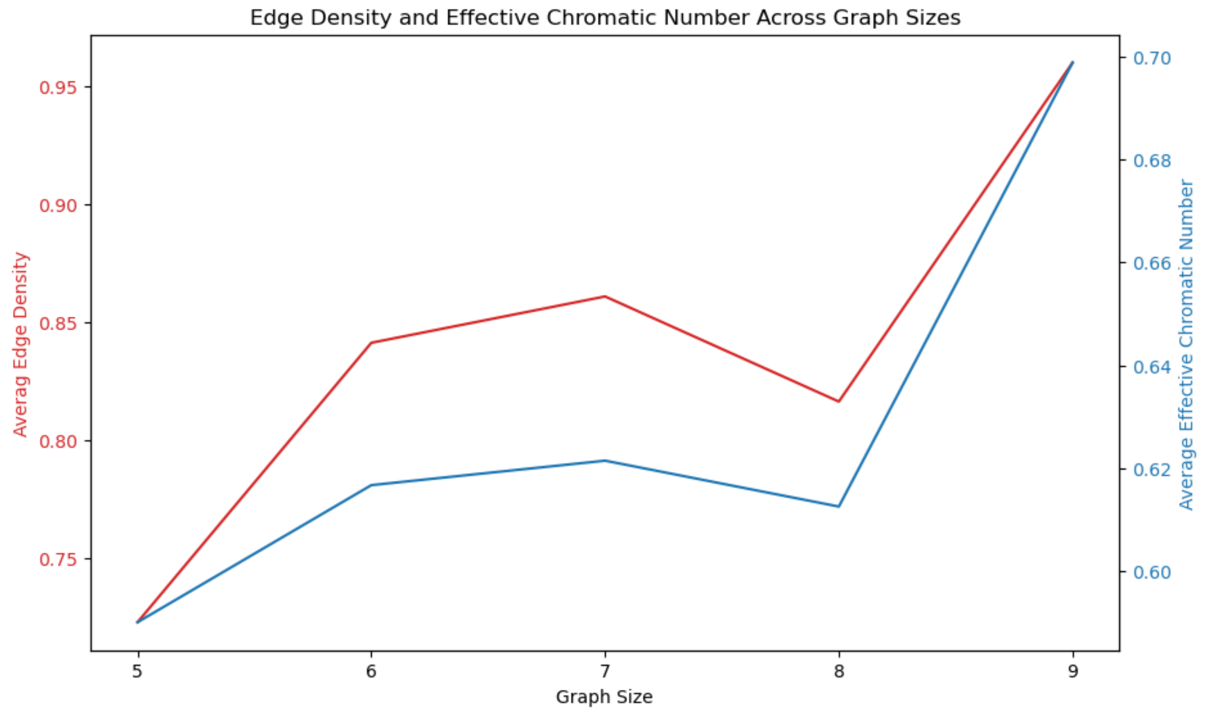




- SOFAI has success rate of 100%, since it will call S2 solver after 5 unsuccessful iterations of the S1 solver
- With graph size 5, SOFAI will return the correct solution just by using the LLM 60% of the times, and calling the S2 solver 40% of the time
- More constraints make it more difficult for the LLM, until (size 9) this is compensated by the available colors

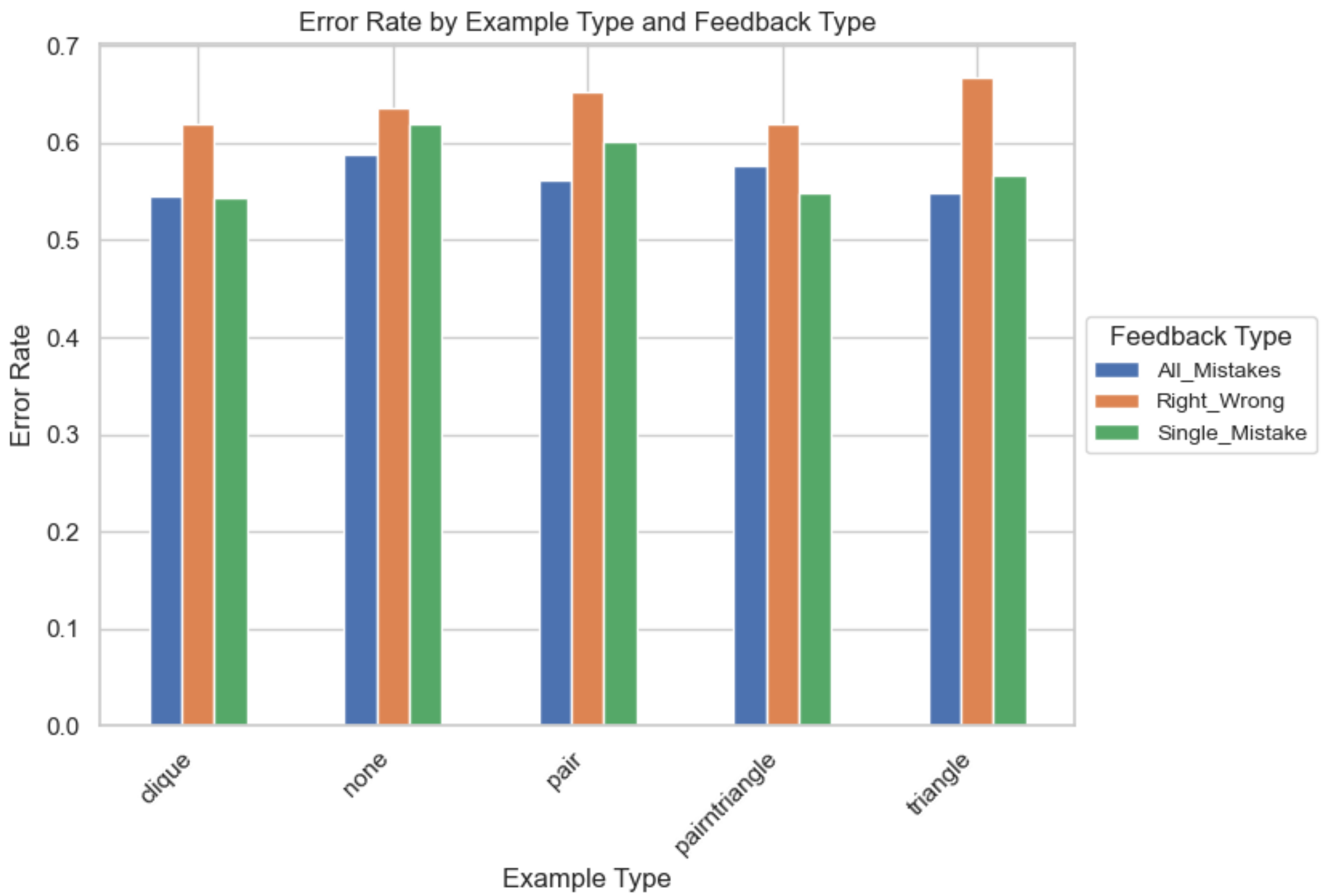
# Understanding the trend





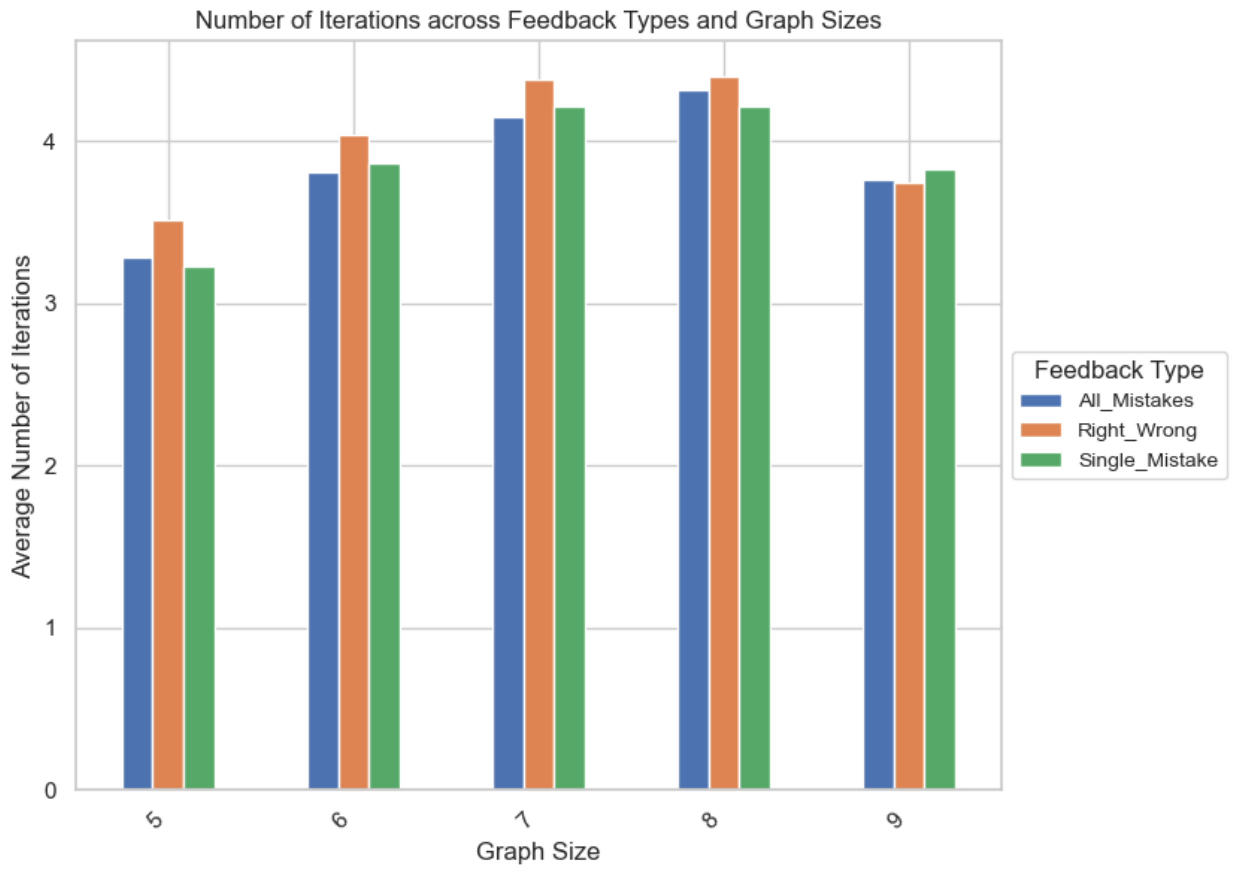
- Graph sizes from 5 to 8: Reduction in both Edge Density and Effective Chromatic Number (less constraints but also less colors)
- Graph size 9: Rise in both Edge Density and Effective Chromatic Number (more constraints but also more available colors)

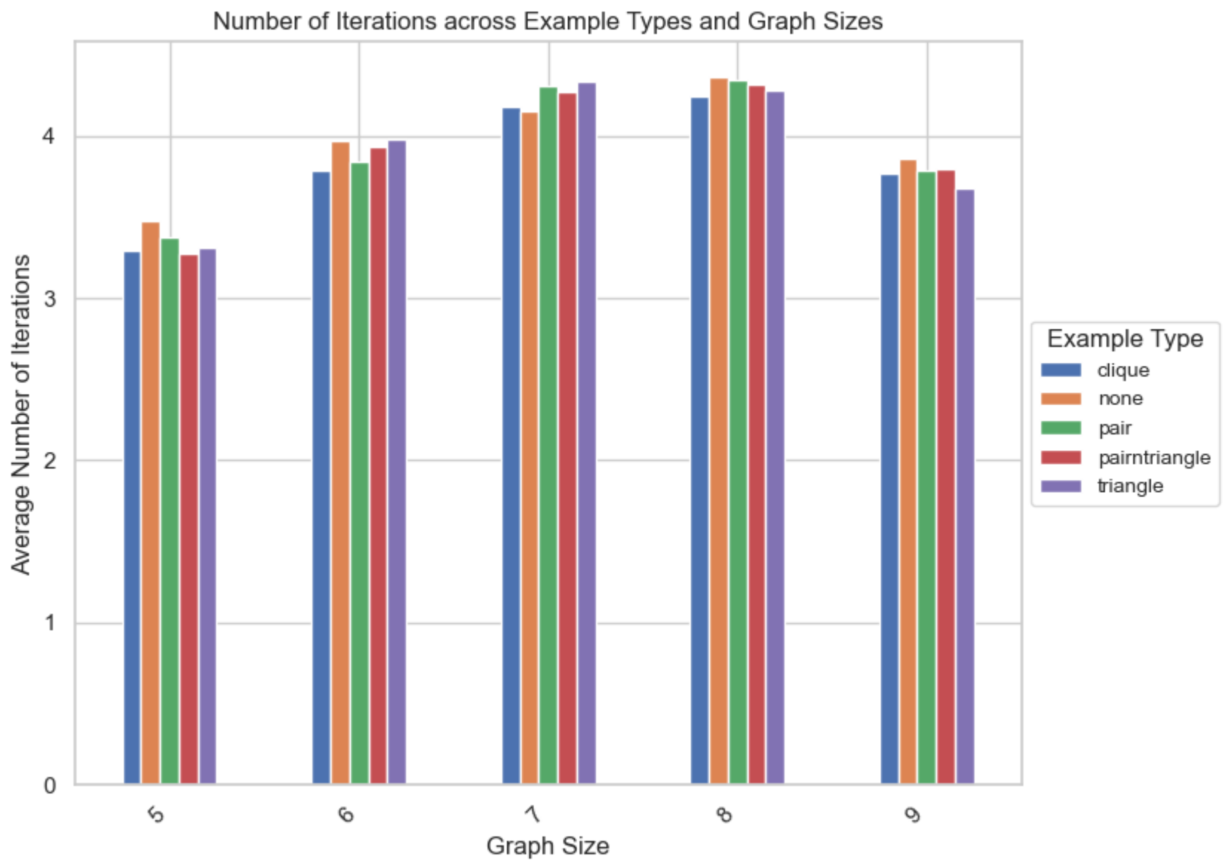
# Error Rate of the S1 solver, by Example type and Feedback Type



Providing all mistakes as feedback helps the LLM most, then single mistake, then just right/wrong.

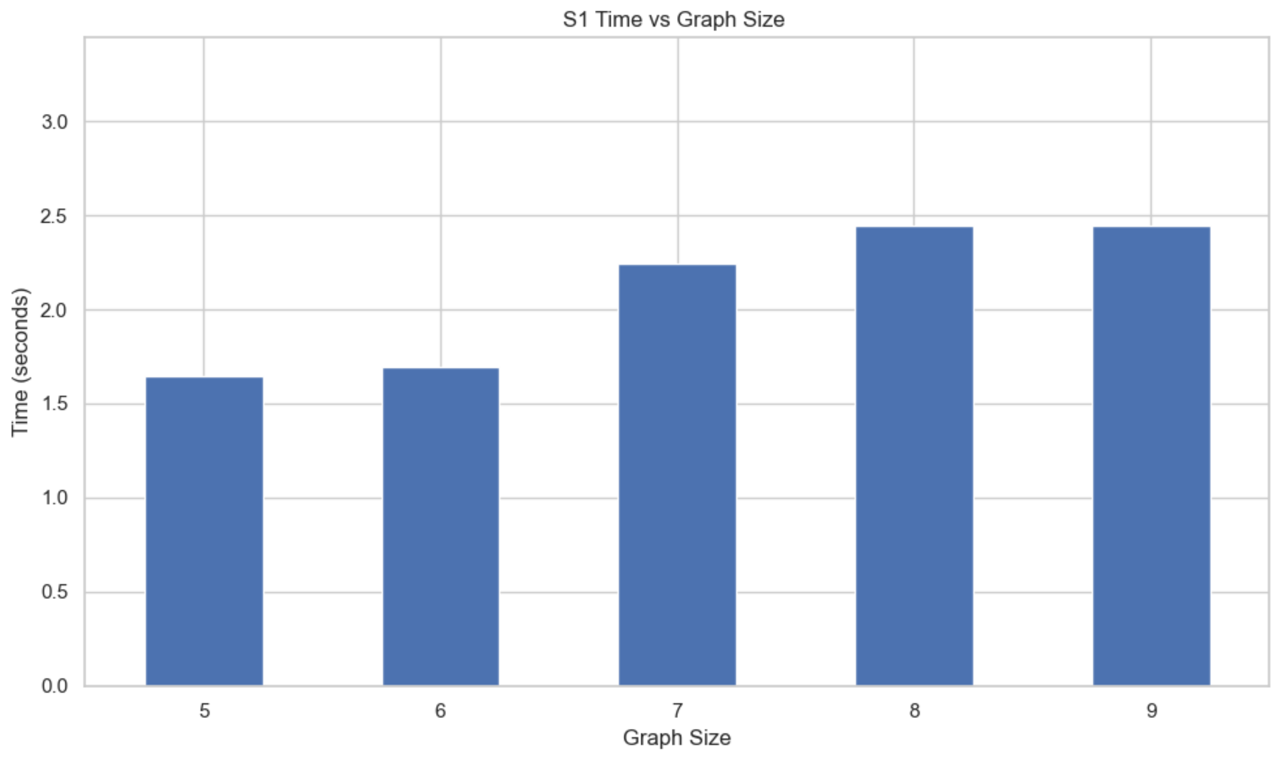
# Number of S1 iterations, by Feedback Type and Example Type

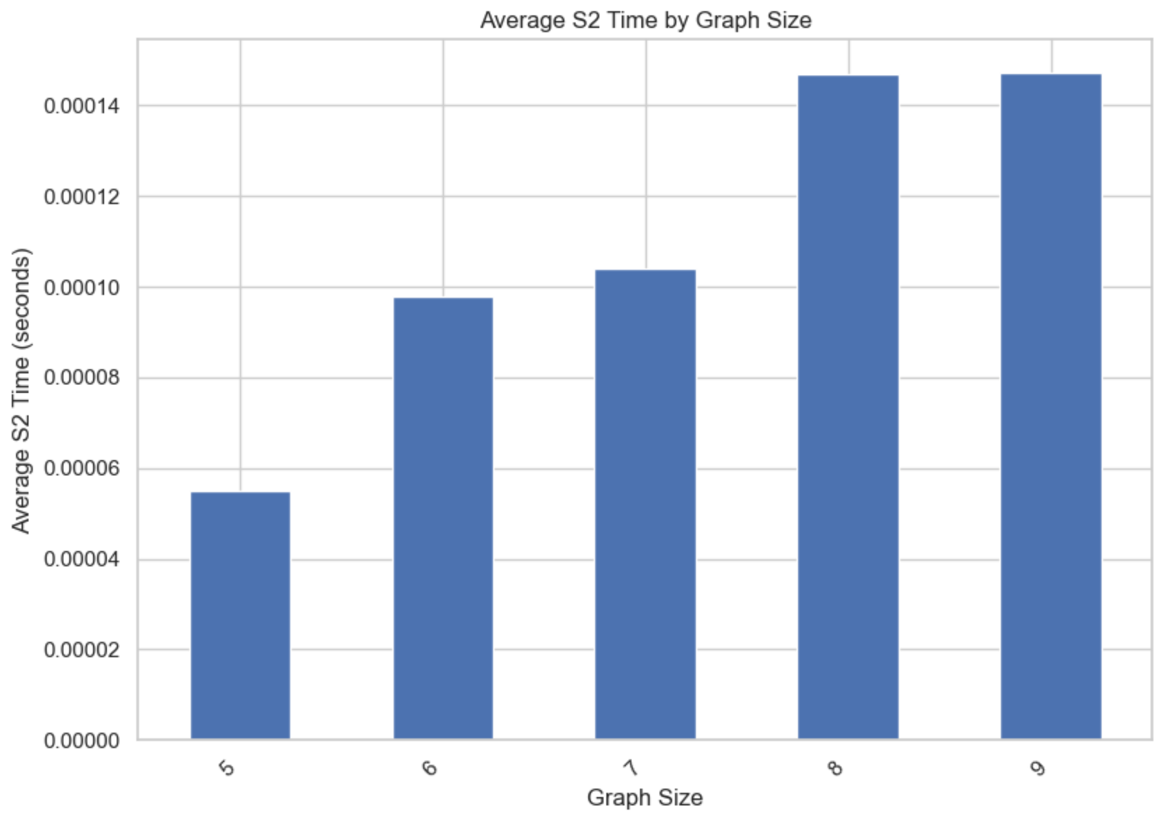




Less iterations if feedback on mistakes, compared to just right/wrong feedback, and more complex examples.

# Time Comparison – S1 vs S2

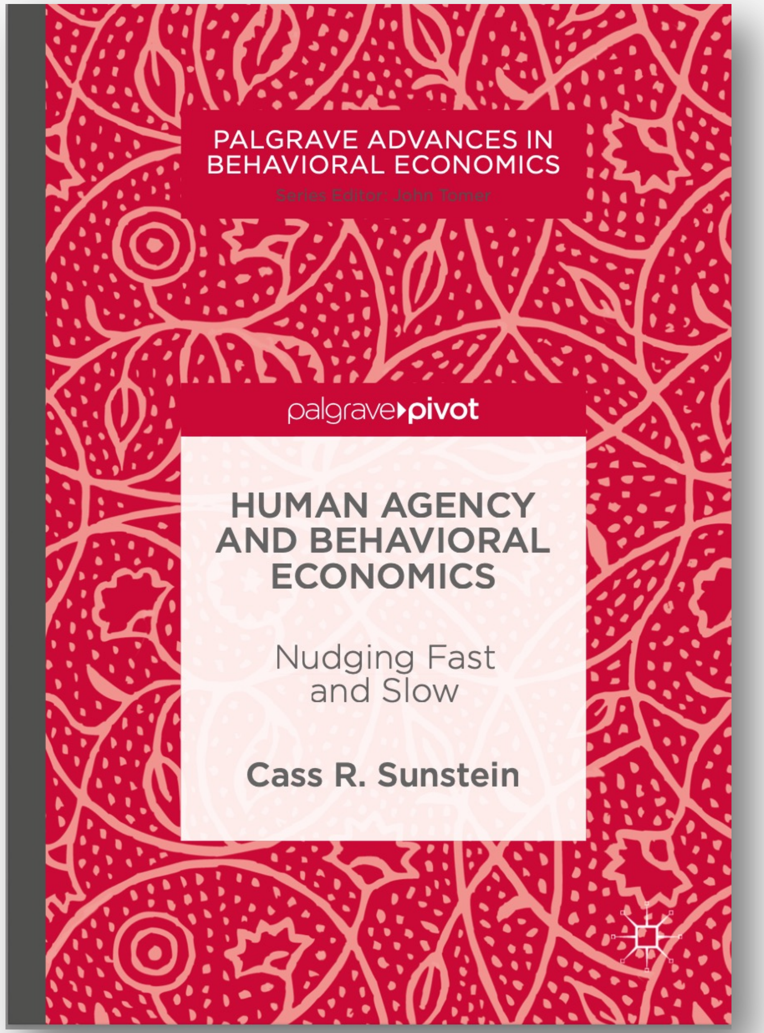




S1 solver time rather stable, S2 solver time grows

## Human decision support: introducing friction when needed to avoid over-trust

- Nudges to use S1: non-educational nudges
    - Ex.: images on cigarette packs
    - Stronger impact on behavior
- Nudges to use S2: educational nudges
    - Ex.: nutrition label
    - Impact mostly on beliefs



PALGRAVE ADVANCES IN BEHAVIORAL ECONOMICS

Series Editor: John Tomer

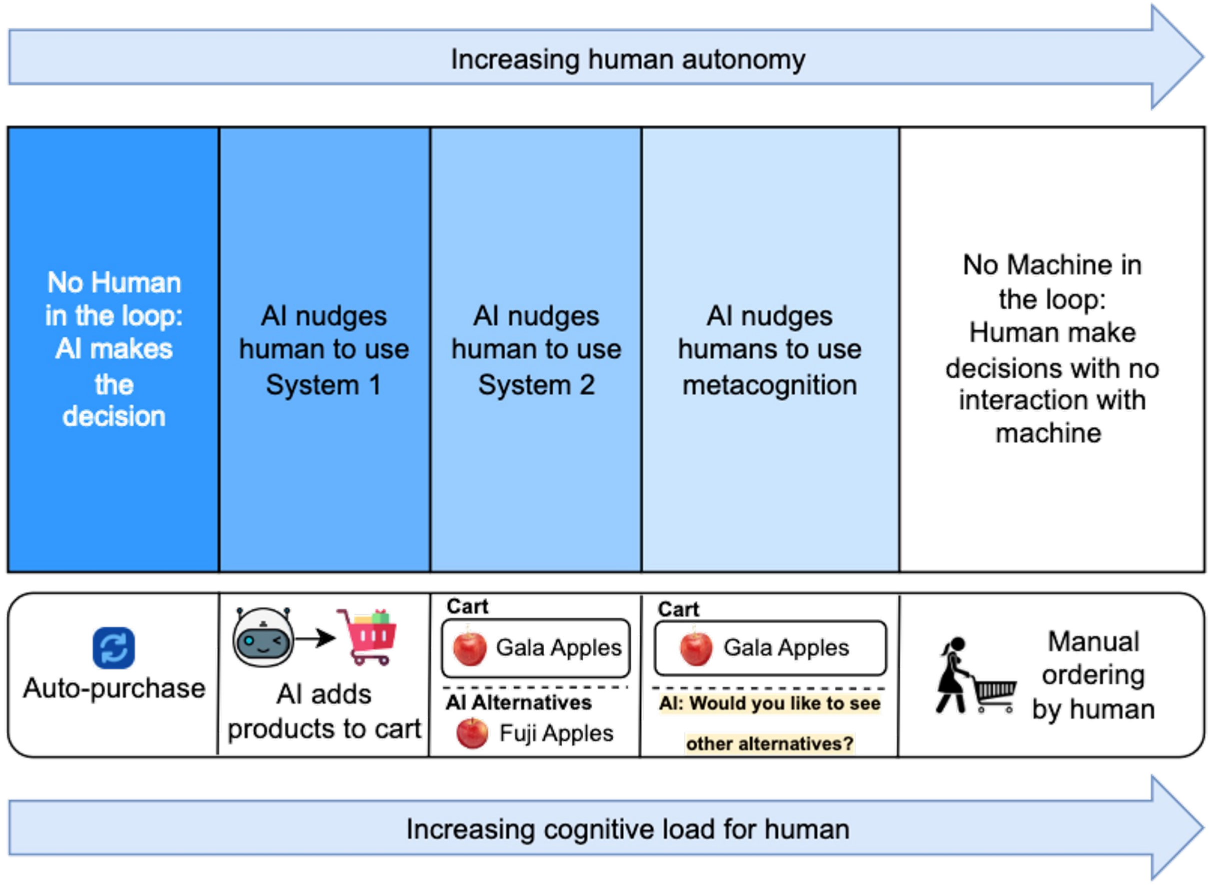
palgrave▸pivot

HUMAN AGENCY AND BEHAVIORAL ECONOMICS

Nudging Fast and Slow

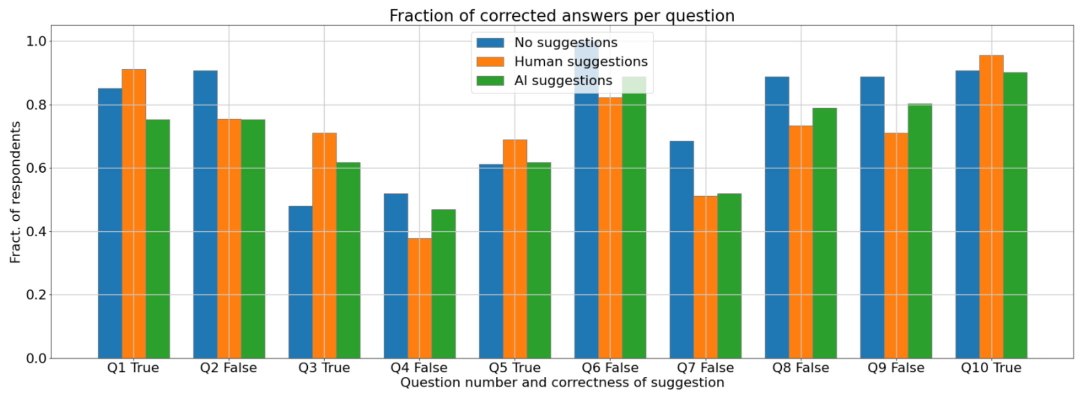
Cass R. Sunstein

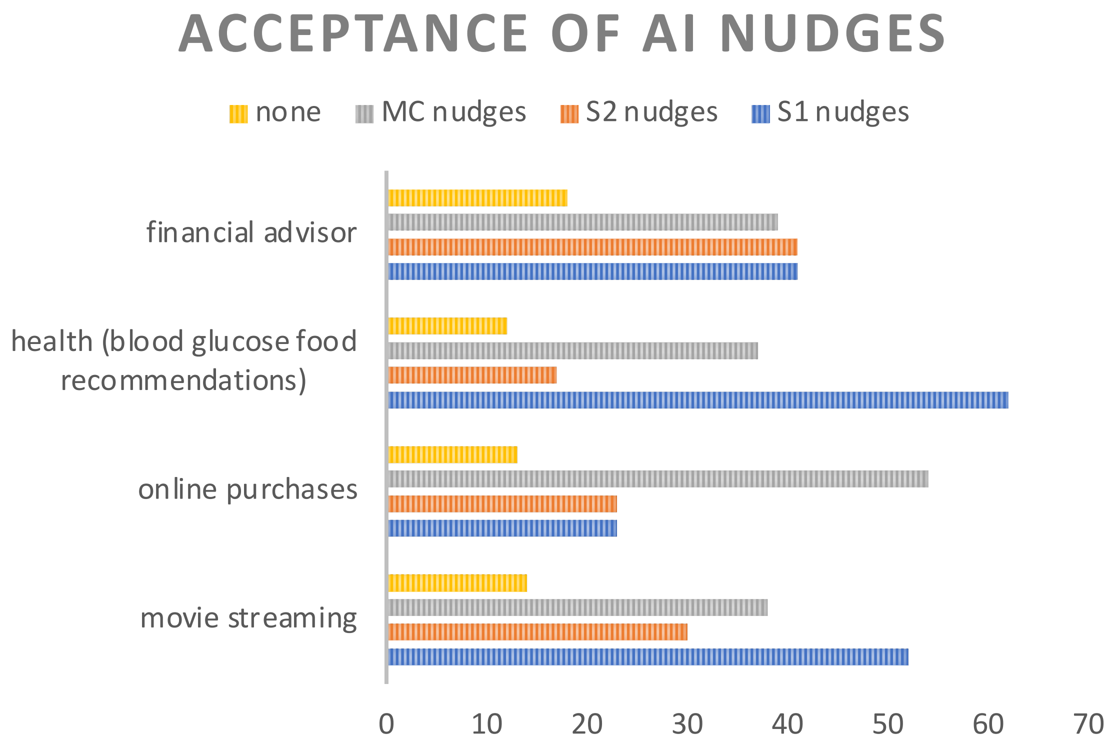
# Human-machine framework: five interaction modalities


Increasing human autonomy
No Human in the loop: AI makes the decision
AI nudges human to use System 1
AI nudges human to use System 2
AI nudges humans to use metacognition
No Machine in the loop: Human make decisions with no interaction with machine
Auto-purchase
AI adds products to cart
Cart
Gala Apples
AI Alternatives
Fuji Apples
Cart
Gala Apples
AI: Would you like to see other alternatives?
Manual ordering by human
Increasing cognitive load for human

# Validating hypothesis and research questions

- **RQ1: Are AI nudges generating a significant anchoring effect?**
  - **Yes**
- **RQ2: Is the AI nudge anchoring effect comparable to the human one?**
  - **Yes, but AI nudging is weaker**
- **RQ3: Are AI nudges accepted by humans as a human-machine collaboration mechanism?**
  - **Yes, by a large majority (at least 82%)**
- **RQ4: Are System 1 AI nudges accepted more or less than System 2 AI nudges?**
  - **It depends on the domain (health, music streaming online purchases, financial wellbeing)**
- RQ5: Are FASCAI's decisions better than humans' alone or AI's alone?
- RQ6: In the FASCAI instance, are humans learning over time, more than if they are asked to make decisions by themselves?
- RQ7: Is human agency preserved in FASCAI?





# Summary and ongoing work

- SOFAI support the combination of S1 and S2 solvers
- The S2 solver helps the S1 solver with examples
- MC helps the S1 solver with feedback
- Working on …
    - S1 solver to help the S2 solver with contextual information
    - Problems for which the S2 solver is very costly
    - Examples generated by S2 solver on smaller problem instances compared to the one that needs to be solved
        - Ex.: in blocksworld planning, put all blocks on the table, or pick just one goal instead of all
    - Open-source SOFAI architecture where one can plug in S1 and S2 solvers, solution evaluators, example generators

# Project Web Site - https://sites.google.com/view/sofai/home

Thinking Fast and Slow in AI

Home

Publications

Panels, Podcasts, and Talks

Scientific Events

Media Coverage

Code

Team

Contact Us

Thinking Fast and Slow in AI

*Leveraging cognitive theories*

*of human decision making to advance AI*

**Scope.** Here we describe the activities and results of a research project whose aim is to advance AI by leveraging cognitive theories of how humans make decisions, with particular focus on the "thinking fast and slow theory" of Daniel Kahneman. The project is led by IBM Research in collaboration with several academic partners.

# Thanks!