A Neurosymbolic Fast and Slow Architecture for Graph Coloring

Vedant Khandelwal[1] VEDANT@EMAIL.SC.EDU
Vishal Pallagani[1] VISHALP@MAILBOX.SC.EDU
Biplav Srivastava[1] BIPLAV.S@SC.EDU
Francesca Rossi[2] FRANCESCA.ROSSI2@IBM.COM
[1]University of South Carolina, Columbia, SC, USA.

[2]IBM Research, NY, USA.

Abstract

Constraint Satisfaction Problems (CSPs) present significant challenges to artificial intelligence due to their intricate constraints and the necessity for precise solutions. Existing symbolic solvers are often slow, and prior research has shown that Large Language Models (LLMs) alone struggle with CSPs because of their complexity. To bridge this gap, we build upon the existing SOFAI architecture (SOFAI-v1), which adapts Daniel Kahneman's "Thinking, Fast and Slow" cognitive model to AI. Our enhanced architecture, SOFAI-v2, integrates refined metacognitive governance mechanisms to improve adaptability across complex domains, specifically tailored here for solving the graph coloring problem, a specific type of CSP. SOFAI-v2 combines a fast System 1 (S1), leveraging LLMs, with a deliberative System 2 (S2), governed by a metacognition module. S1's initial solutions, often limited by constraint adherence issues, are improved through targeted feedback and examples from metacognition, aligning S1 more closely with CSP requirements. If S1 fails to resolve the problem, metacognition strategically invokes S2, ensuring accurate and reliable solutions. Our empirical results demonstrate that SOFAI-v2 achieves a 10.5% higher success rate and is up to 30% faster than a traditional symbolic solver in solving graph coloring problems.

1. Introduction

Constraint Satisfaction Problems are a core challenge in artificial intelligence (AI) due to their demand for correctness adhering to strict constraints in static (Kumar, 1992) or uncertain, dynamic environments (Verfaillie & Jussien, 2005). These problems are prominent in applications such as scheduling, boolean satisfiability, resource allocation, temporal reasoning, and planning (Welsh & Powell, 1967; Chaitin, 1982). Traditional approaches to solve CSPs typically rely on constraint propagation or search (Haralick & Elliott, 1980). The symbolic approaches are known for their accuracy but often struggle with scalability as the complexity of CSPs increases, hindered by high computational overhead and slow processing times (Dechter, 2003; Rossi et al., 2006).

Recent advancements in LLMs have shown promise in rapidly processing complex information (Ruoss et al., 2024), but they are fundamentally limited in solving sequential decision-making tasks like planning (Valmeekam et al., 2022), and reasoning (Stechly et al., 2024). Due to their probabilistic, retrieval-based nature, LLMs approximate solutions rather than generating definitive answers, often leading to partial or inconsistent adherence to constraints (Jiang et al., 2023). Even

This work is licensed under a Creative Commons Attribution International 4.0 License.

with state-of-the-art prompting strategies and self-verification techniques, LLMs fall short in handling strict requirements of CSPs, making their outputs unreliable for applications demanding high precision. We study feasibility under fixed resource limits in the graph coloring decision problem, a core constraint satisfaction setting that supports clear verification and iterative guidance.

The disparity between the precision of traditional symbolic methods and the adaptability and speed of LLMs underscores the need for a new paradigm in tackling complex tasks like graph coloring (Jensen & Toft, 2011), a canonical CSP. Drawing inspiration from Daniel Kahneman's cognitive theory of "Thinking, Fast and Slow" (Kahneman, 2011) and its adaptation to AI as SOFAI-v1 (Fabiano et al., 2025; Booch et al., 2021), we introduce SOFAI-v2 which extends the earlier architecture by refining metacognitive mechanisms that oversee a fast, experience-based S1, powered by an LLM, and a slow, deliberative S2 using the DSATUR algorithm for graph coloring. This governance mechanism continuously monitors and enhances S1's outputs, offering targeted feedback and constraint adherence examples; when S1's solutions fall short, S2 is strategically invoked to guarantee accuracy. In this setup, S1 swiftly produces initial solutions, while S2 provides a reliable fallback for achieving precise constraint satisfaction. This work studies problem solving with a fast S1, a deliberate S2, and an MC that monitors and guides iterations, with memory for reuse. Beyond the S1/S2 framing, SOFAI-v2 is positioned as a *neurosymbolic* cognitive architecture: S1 (neural) proposes, symbolic constraints verify, and MC enacts control. This aligns with computational accounts of automatic versus controlled processing and supervisory control (e.g., contention scheduling and supervisory attention) (Norman & Shallice, 1986; Cooper & Shallice, 2000), providing a bridge from cognitive theory to an AI system.

The implemented system targets the graph coloring decision task and evaluates success rate, time, and iteration behavior. Our contributions are as follows:

- Introduced SOFAI-v2, a neurosymbolic fast and slow architecture to enhance the performance of CSP solvers on graph coloring problems through adaptive metacognitive governance.
- Created a comprehensive benchmark of graph coloring problems, featuring variations in graph size, edge probability, and solvable and unsolvable instances to test adaptability.
- Conducted extensive empirical evaluations of various solver configurations for graph coloring, demonstrating advantages of SOFAI-v2 in terms of success rate, time efficiency.

The rest of the paper is organized as follows: we first provide the necessary background and discuss prior work. Next, we present the SOFAI-v2 architecture, followed by the experimental setup and results. Finally, we conclude with a summary of our findings.

2. Background

2.1 Graph Coloring Decision Problem

The graph coloring decision problem is a specific type of constraint satisfaction problem (CSP), where the objective is to determine whether a given graph can be colored using at most k colors such that no two adjacent nodes share the same color. Formally, the problem is defined as follows:

- An undirected, unweighted graph $G = (V, E)$
- A positive integer k representing the maximum number of colors
- Determine whether there exists a function $f : V \to \{1, 2, \ldots, k\}$ such that for every edge $(u, v) \in E$, $f(u) \neq f(v)$.

Unlike the classical graph coloring problem, which aims to minimize the number of colors used (an optimization problem), the decision version focuses solely on satisfiability: whether a valid coloring exists for a given k. We use the DIMACS representation (Johnson & Trick, 1996) to describe the graph coloring problem. The decision problem asks a clear yes/no question for a fixed k. This matches many practical settings where a fixed resource limit is given and the goal is feasibility, e.g., schedules with a fixed number of time slots, register allocation with a fixed number of registers, or frequency assignment with a fixed set of bands. It provides MC with a simple accept/reject signal and makes it straightforward to generate targeted feedback when constraints fail, which we interpret as a cue for supervisory control in the architecture (Norman & Shallice, 1986; Cooper & Shallice, 2000).

2.2 Degree of Saturation Algorithm with Backtracking

The Degree of Saturation (DSATUR) algorithm assigns colors to graph vertices based on their saturation degree, prioritizing vertices that are the most constrained—those having adjacent vertices with the greatest variety of colors already assigned (Brélaz, 1979). DSATUR remains competitive for graph coloring due to its heuristic efficiency, especially for sparse and mid-sized graphs, and its direct operation on graph structures without requiring transformations typical of SAT solvers. Recent studies (Yekezare et al., 2024; Schidler & Szeider, 2023) highlight its effectiveness, making it a suitable choice for our symbolic solver. However, the traditional DSATUR algorithm targets the optimization version of the graph coloring problem, minimizing the number of colors. For our work, we adapt DSATUR into a satisfiability solver by constraining it with a fixed maximum number of colors. A detailed description of the modified algorithm is provided in Supplementary Material (Khandelwal et al., 2024). Unlike other solvers, DSATUR can be straightforwardly modified for the decision version of the problem, making it a practical and adaptable choice for our symbolic backend. In SOFAI-v2, DSATUR with backtracking plays the role of slow, controlled processing that is selectively invoked by MC under conflict or low confidence, consistent with supervisory control accounts in cognitive systems (Norman & Shallice, 1986; Cooper & Shallice, 2000).

3. Related Works

This section reviews the evolution of solvers for graph coloring, the application of LLMs in sequential decision-making tasks, and the emergence of neurosymbolic approaches for enhancing decision-making processes. *We also situate SOFAI-v2 within cognitive control theories that distinguish routine (automatic) processing from controlled, supervised processing (Norman & Shallice, 1986; Cooper & Shallice, 2000).*

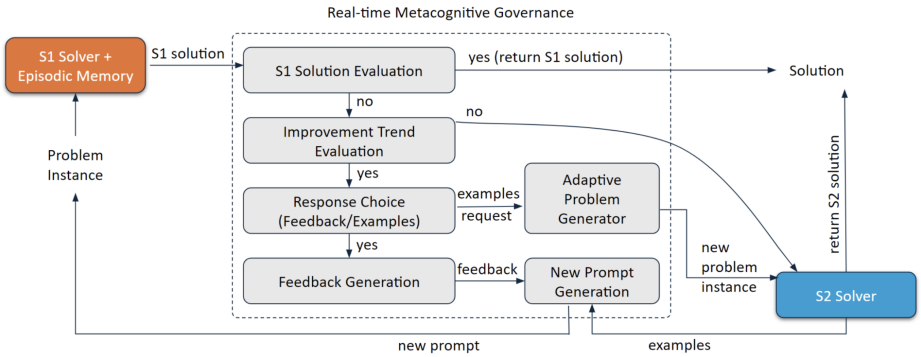


Figure 1: SOFAI-v2 architecture with real-time metacognitive governance (supervisory control over fast/automatic and slow/controlled processing)

3.1 Solvers for Graph Coloring

Traditional symbolic solvers have been extensively developed for this problem, leveraging algorithms that systematically explore the search space. Early works like (Brélaz, 1979) introduced efficient heuristics such as the DSATUR algorithm, significantly improving practical performance. (Chaitin, 1982) applied graph coloring to compiler optimization, demonstrating the versatility of symbolic methods. (Golumbic, 2004) provided a comprehensive treatment of graph algorithms, including coloring techniques. Despite their accuracy, symbolic approaches often struggle with scalability as CSP complexity increases, hindered by high computational overhead and slow processing times (Dechter, 2003). Techniques like constraint propagation (Mackworth, 1977) and backtracking algorithms (Bitner & Reingold, 1975) have been employed to enhance efficiency but are limited by the exponential growth of the search space.

DSATUR selects at each step the uncolored vertex with the highest *saturation degree* (the number of distinct colors among its colored neighbors), breaking ties by degree (Brélaz, 1979). This ordering aggressively focuses search on the most constrained vertices, yielding strong practical performance. With appropriate data structures, per-step selection can be done in near $O(logn)$ update time, but the overall search with backtracking remains exponential in the worst case due to the NP-completeness of graph coloring. In our setting, we couple DSATUR with backtracking under a fixed color budget k: the procedure attempts to extend a partial assignment using at most k colors; failure to extend yields a certified UNSAT outcome, while success produces a constructive k-coloring witness.

To address these limitations, neural solvers have been explored, utilizing machine learning techniques to approximate solutions for graph coloring. For instance, (Li et al., 2018) proposed using graph neural networks to tackle combinatorial optimization problems. Similarly, (Khalil et al., 2017) introduced a framework that learns heuristics directly from data, improving scalability. However, neural approaches often face challenges in strictly adhering to the constraints inherent in CSPs. In contrast, our SOFAI setup directly targets the *decision* form of graph coloring with an explicit k, integrating DSATUR+backtracking as S2 and using S1/SOFAI components to propose assignments that are subsequently verified. Many end-to-end neural solvers are not readily adaptable to this

satisfiability variant, which limits their comparability to our setting; we therefore retain them here primarily for historical context and contrast, while focusing our technical discussion on DSATUR and its role in our pipeline.

3.2 LLMs for Reasoning Tasks

Recent developments in LLMs have opened new avenues for addressing sequential decision-making tasks, including CSPs. Various prompt engineering techniques have been proposed to enhance the reasoning capabilities of LLMs (Brown, 2020; Wei et al., 2022; Kojima et al., 2022; Nye et al., 2021; Wang et al., 2022). (Han et al., 2023) demonstrated that LLMs can arrive at correct answers for graph-based problems through iterative prompting. However, the reasoning capabilities of LLMs have been found to be fundamentally approximate retrieval in nature Kambhampati (2024), limiting their effectiveness in precise reasoning tasks. This limitation is evidenced by studies showing LLMs' inability to solve planning problems (Valmeekam et al., 2022) and graph coloring problems (Stechly et al., 2024), where strict adherence to constraints is required (Jiang et al., 2023). *In cognitive-systems terms, LLM-only routines resemble contention among habitual schemas without sufficient supervisory control (Norman & Shallice, 1986).*

3.3 Neurosymbolic Approaches for Reasoning Tasks

Neurosymbolic approaches have gained considerable attention in addressing complex sequential decision-making tasks such as automated planning (Fabiano et al., 2023), constrained grid navigation (Ganapini et al., 2022), and puzzle solving (Lin et al., 2024). These studies demonstrate that integrating neural methods, such as LLMs, with symbolic solvers enhances performance on sequential decision-making tasks. To the best of our knowledge, neurosymbolic techniques have not been explored in the context of graph coloring problems. To our knowledge, prior neurosymbolic studies have not targeted the graph coloring decision problem, nor integrated meta-level feedback with a decision-oriented S2; SOFAI-v2 explicitly addresses this gap. *Conceptually, SOFAI-v2 instantiates a neurosymbolic cognitive architecture in which S1 (neural) handles routine proposals while MC+S2 provide supervisory/controlled processing aligned with Norman–Shallice-style accounts (Norman & Shallice, 1986; Cooper & Shallice, 2000).*

4. The SOFAI Architecture

The SOFAI architecture, also referred to as SOFAI-v1 here (Booch et al., 2021), leverages a rule-based metacognitive control mechanism that dynamically chooses between S1 and S2 solvers based on a confidence threshold associated with S1's outputs. S1 solvers are experience-based (and usually data-driven) solvers, while S2 solvers are deliberative (and usually symbolic and rule-based) solvers. SOFAI-v1 has been instantiated to planning (Fabiano et al., 2023) and constrained-grid navigation (Ganapini et al., 2022), showing improvement on the performance compared to either symbolic or data-driven solvers for the same class of problems.

In SOFAI-v1, the metacognitive component only chooses between S1 and S2 solvers, but does not exploit the possible collaboration between these two kinds of solvers, nor does it provide feed-

back to the solvers if they cannot solve the given problem instance. In this paper we consider a generalized version of this architecture, called SOFAI-v2, shown in Figure 1, where the metacognitive governance (called MC) can provide feedback and examples to a failing S1 solver (that for this paper is an LLM), calling the S1 solver more than once, until either a correct solution is returned or a maximum number of iterations is reached, which triggers the activation of an S2 solver. *This implements a neurosymbolic control loop: S1 produces routine proposals (automatic), MC performs verification and trend monitoring (supervisory attention), and S2 executes controlled search when conflicts persist, echoing contention scheduling with a supervisory system (Norman & Shallice, 1986; Cooper & Shallice, 2000).*

Formally, let $f_{\text{S1}}(x, M)$ denote the solution produced by S1 for a problem instance x with access to an episodic memory M. Here, M consists of previously encountered problem-solution pairs generated by SOFAI-v2, i.e., pairs $(x_j, f_{\text{SOFAI-v2}}(x_j))$, where $f_{\text{SOFAI-v2}}(x_j)$ represents the final solution that could have been generated by either S1 or S2. In the context of this paper, x corresponds to a graph coloring decision problem instance defined by a graph G and a color bound k.

When a new problem instance x is presented to S1, a similarity function $\sigma(x, M)$ is used to retrieve a subset $M_x \subset M$ of similar past instances, defined as:

$$M_x = \{(x_j, f_{\text{SOFAI-v2}}(x_j)) \mid \sigma(x, x_j) \geq \alpha\},$$

where $x_j \in M$ and α is a predefined similarity threshold. This subset, M_x, provides S1 with additional contextual information from past solutions, enabling it to leverage SOFAI-v2's previous outputs for experience-guided context.

The correctness of solution for the graph coloring decision problem can be checked and evaluated in polynomial time by MC. The correctness score used by MC for graph coloring solutions is:

$$C(f_{S1}(x, M_x)) = \frac{\sum_{(v,u)\in E} \mathbf{1}[f(v) \neq f(u)]}{|E|}$$

where:

- $f(v)$ denotes the assigned color of vertex v,
- E denotes the set of edges in the graph.

If this correctness meets or exceeds a threshold θ_{S1}, solution is accepted. Otherwise, MC iteratively provides structured feedback $\mathcal{F}(f_{S1}(x, M_x))$ or simplified examples $\mathcal{E}(x)$. We set the correctness threshold to 1, since 1 indicates full constraint adherence for the graph-coloring decision task; this ensures S1 outputs are only accepted when a complete, validator-checked coloring is produced. MC also monitors S1's solution correctness across iterative feedbacks, identifying improvement trends defined by:

$$\forall i \in \{1, \ldots, m-1\}, \quad C_{i+1} > C_i$$

Failure to show improvement within m iterations results in MC invoking the fallback S2 solver, ensuring robustness. Thus, the iterative feedback mechanism and episodic memory advances SOFAI-v2 over SOFAI-v1, enhancing solver effectiveness via structured feedback and experience-based

learning. Our theoretical commitments are: (i) dual-solver with metacognitive governance, (ii) episodic memory for similarity-based retrieval, and (iii) verify-and-feedback loops grounded in polynomial-time checks. The concrete instantiation (Mistral-7B as S1, DSATUR with backtracking as S2) is an implementation detail; same principles transfer to other domains. Detailed algorithm for the architecture is in Supplementary Material (Khandelwal et al., 2024).

5. Experimental Setup and Results

This section details methodology and experimental results evaluating various solver configurations on graph coloring problems, emphasizing efficiency and success rate under systematic constraints. *We also report how iterative feedback and selective escalation can be interpreted as supervisory control improving routine policy performance under increasing constraint conflict (Norman & Shallice, 1986).*

5.1 Data Generation and Problem Classification

Graph coloring instances were generated using the Erdős–Rényi model, controlling problem complexity through three primary parameters:
Graph Size (n): Varies between $n \in [5, 50]$ vertices, encompassing a diverse spectrum of problem sizes.
Edge Probability (p): Adjusted within $p \in [0.1, 0.9]$, influencing edge density and problem difficulty.
Solvability Mix (m): The chromatic number ($\chi(G)$) of each generated graph was computed using the DSATUR algorithm. Problems were then categorized based on solvability, represented as $m = (a, b)$, where a and b denote percentages of solvable and unsolvable instances respectively:

- $m = (100, 0)$: All problems solvable within constraints.
- $m = (50, 50)$: Equal distribution of solvable and unsolvable problems.
- $m = (0, 100)$: All problems unsolvable; set by choosing the number of available colors $k < \chi(G)$, ensuring infeasibility.

For each graph size and edge probability pair 100 graph problems were generated and stored in standardized DIMACS format for reproducibility. *Github Repository:* `https://github.com/khvedant02/CSP-SOFAI_v2` .

5.2 SOFAI-v2 Implementation for Graph Coloring

SOFAI-v2 integrates iterative feedback mechanisms and episodic memory to dynamically resolve graph coloring problems. Figures 2 and 3 illustrate example prompts.

- **Episodic Memory (M):** Stores historical graph instances with attributes—vertices, edges, edge density, chromatic number—and solutions. A similarity function $\sigma(x, M)$ retrieves relevant past solutions:

$$M_x = \{(x_j, f_{\text{v2}}(x_j)) : x_j \in M,\ d(x, x_j) < \alpha\}$$

where $d(x, x_j) = |attributes(x) - attributes(x_j)|$, and α is a similarity threshold.

- **Iterative Feedback Mechanism:** If the solver's correctness score (defined in Section 4) is insufficient, a two-step feedback process occurs:
  1. *Detect Conflict Pairs*: Identify adjacent vertices assigned identical colors.
  2. *Template-Based Feedback*: Algorithmically generate structured corrections (e.g., "Error: Vertices A and B are adjacent but have the same color"). Example of feedbacks is given in Supplementary Material (Khandelwal et al., 2024).
- **Improvement Evaluation and S2 Invocation:** If no solution improvement occurs over five iterations or if correctness remains below threshold, S2 is invoked as a fallback. If either solver identifies an instance as unsolvable, the instance is labeled as UNSAT. Exceeding the strict 200-second per-instance time limit results in failure, regardless of actual solvability.
- **Example Generation ($\mathcal{E}(x)$):** A greedy-based algorithm generates simpler subgraph instances to guide S1. Examples are detailed further in Supplementary Material (Khandelwal et al., 2024).

This comprehensive integration ensures SOFAI-v2 leverages both real-time metacognitive governance and experience-guided context for enhanced problem-solving effectiveness. *Operationally, this mirrors a supervisory system that monitors conflict and recruits controlled processing when routine proposals fail (Norman & Shallice, 1986; Cooper & Shallice, 2000).*

SOFAI-v1 vs SOFAI-v2: SOFAI-v2 introduces substantial advancements over previous works of SOFAI (SOFAI-v1), by incorporating iterative feedback mechanisms and episodic memory, which enhance the problem-solving capabilities of the S1 solver. Unlike SOFAI-v1, where problems not solved by S1 are directly escalated to S2 without iterative refinement, SOFAI-v2 systematically leverages historical data and adaptive feedback to optimize solutions before resorting to S2, thereby improving efficiency and efficacy. Supplementary Material (Khandelwal et al., 2024) includes exact prompts, feedback templates, and our DSATUR+BT pseudocode.

5.3 Performance Metrics and Evaluation

All solver configurations (Table 1) were evaluated under a strict 200-second time limit per instance. Solver performance was measured using:

$$\text{Success Rate}(\%) = \left(\frac{\text{Number of Correct Solutions}}{\text{Total Number of Problems}}\right) \times 100 \tag{1}$$

$$\text{Average Time Taken} = \frac{\sum_{i=1}^{n} t_i}{n} \tag{2}$$

where t_i is the solving time for the i-th successfully solved instance and n the total instances solved within the time limit. Instances exceeding this limit are marked as failures, explaining occasional 0% success rates even when solvable theoretically. Results reported are averaged over three randomized trials for reliability. To better understand solver efficiency beyond strict time constraints,

Graph Coloring Problem Prompt

New Problem to Solve:
You are given an undirected graph with 2 colors available. Your task is to assign a color to each vertex such that no two adjacent vertices share the same color.

Graph Representation:
- Number of vertices and edges: `p edge 5 5`.
- Edges between vertices are listed as follows:

```
e A B
e A C
e B C
e C D
e D E
```

Objective:
Assign a unique color to each vertex, ensuring that no two vertices connected by an edge have the same color. Use no more than 2 distinct colors. Provide the color assignments for each vertex in the format:
(Vertex Color)

Example Format:

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

Please provide the color assignment for the new problem to solve, or respond with "NOT SOLVABLE" if it cannot be solved.

Figure 2: LLM prompt template for the graph coloring decision problem without using episodic memory.

we conducted supplementary experiments without time limits, solely for comparative runtime analysis. *From a cognitive perspective, increased success with iterative MC reflects benefits of supervisory control over routine responding under higher constraint densities (Norman & Shallice, 1986; Cooper & Shallice, 2000).*

5.4 Results

This section presents an analysis of solver performance by systematically controlling key parameters. Unless mentioned, the edge formation probability is fixed to $p = 0.5$, allowing a focused evaluation across varying graph sizes n and solvability mixes m to capture a broad range of problem configurations. Tables 2 and 3 provide the success rate and average time taken for each solver

Episodic Memory Prompt

Problem:

```
p edge 4 4
e X Y
e Y Z
e Z A
e X A
```

Correct Solution:

```
(X 1)
(Y 2)
(Z 1)
(A 2)
```

End of Example

Figure 3: Example for episodic memory, which is added at the end of LLM prompt template in Supplementary Material (Khandelwal et al., 2024).

Table 1: Descriptions of solver configurations employed in experiments.

ID	Description
S1	Mistral-7B LLM
S2	DSATUR with Backtracking
SOFAI-v1	Combines S1 and S2 with rule-based MC
SOFAI-v2	Combines S1 and S2 with iterative MC
MC-S1-I[1-5]	Metacognitive feedback (1–5 iterations) to refine S1 solutions

configuration under these controlled conditions. The Supplementary Material (Khandelwal et al., 2024) includes results for additional p values to supplement this focused analysis, providing a comprehensive view of solver robustness across varying edge densities. From these results, we aim to address four key research questions (RQs) that provide deeper insights into SOFAI-v2 performance and robustness.

RQ 1

How does SOFAI-v2 success rate compare to that of S1, S2, SOFAI-v1?

Answer to RQ 1: SOFAI-v2 demonstrates substantial improvements in success rates over other solvers, particularly in challenging problem configurations involving unsolvable or mixed solvability instances. Table 2 shows that for clearly solvable scenarios ($m = (100, 0)$), SOFAI-v2 matches

Table 2: Success rates (%) across solvers and problem configurations for graph sizes from 5 to 50 with edge probability $p = 0.5$. Configurations $m = (100, 0)$, $m = (0, 100)$, and $m = (50, 50)$ represent solvable, unsolvable, and a balanced mix of solvable and unsolvable instances, respectively.

Graph Size	m(100,0) [%]				m(0,100) [%]				m(50,50) [%]			
	S1	S2	SOFAI_v1	SOFAI-v2	S1	S2	SOFAI_v1	SOFAI-v2	S1	S2	SOFAI_v1	SOFAI-v2
5	9.41	**100**	**100**	**100**	75	**100**	**100**	**100**	45.36	**100**	**100**	**100**
10	0	**100**	**100**	**100**	60.38	**100**	**100**	**100**	39.39	**100**	**100**	**100**
15	0	**80**	**80**	**80**	37.50	77.08	89.58	**93.75**	17.65	77.45	83.33	**84.31**
20	0	5	5	**7.29**	45.65	4.35	47.83	**76.09**	22.58	3.06	24.73	**38.95**
30	0	0	0	0	54.17	0	62.50	**72.92**	28.89	0	33.71	**38.89**
40	0	0	0	0	33.33	0	**47.37**	**47.37**	17.43	0	24.55	**24.77**
50	0	0	0	0	3.85	0	3.85	**53.85**	2.11	0	2.11	**27.72**

Table 3: Average time (seconds) for SOFAI_v1, SOFAI_v2, and S2 across graph sizes and configurations $m(100, 0)$, $m(0, 100)$, and $m(50, 50)$.

Graph Size	$m(100, 0)$ (s)			$m(0, 100)$ (s)			$m(50, 50)$ (s)		
	SOFAI_v1	SOFAI_v2	S2	SOFAI_v1	SOFAI_v2	S2	SOFAI_v1	SOFAI_v2	S2
5	2	**1**	0	2	**1**	0	2	**1**	0
10	4	**2**	0	4	**2**	0	4	**3**	0
15	50	**40**	51	32	**19**	59	52	**47**	54
20	87	**65**	82	50	**20**	97	87	**84**	107
30	323	**306**	320	403	**299**	400	353	**263**	350
40	5043.2	**4512.3**	4995.8	5006.9	**3511.7**	4989.4	5012.1	**3524.6**	4992.7
50	20087.3	**18034.9**	19950.5	20044.2	**14102.7**	19920.6	20051.8	**14178.3**	19965.0

the optimal performance of SOFAI-v1 and S2 across smaller graph sizes. However, its key advantage becomes apparent in more complex scenarios:

- For unsolvable scenarios ($m = (0, 100)$) at graph size 50, SOFAI-v2 achieves a success rate of 53.85%, far surpassing the performance of SOFAI-v1 and S1, which each attain only 3.85%. This translates to an improvement of approximately 1298%.
- In mixed scenarios ($m = (50, 50)$), SOFAI-v2 maintains a success rate of 27.72%, substantially higher compared to just 2.11% by SOFAI-v1 and S1.

These performance gains arise primarily from SOFAI-v2's enhanced metacognitive governance, which systematically leverages iterative solver feedback and episodic memory. This adaptive capability allows SOFAI-v2 to dynamically resolve constraints and effectively handle complex or initially challenging instances that other solvers cannot reliably solve.

For the above results we set a fixed time limit of 200 seconds per problem. If a solver does not finish within this time, it is counted as a failure. The average time taken results for this setting is reported separately in the Supplementary Material (Khandelwal et al., 2024).

RQ 2

How does SOFAI-v2 compare in terms of average time relative to SOFAI-v1 and S2?

Answer to RQ 2: SOFAI-v2 demonstrates substantial improvements in time efficiency compared to SOFAI_v1 and S2, especially for complex configurations involving unsolvable or mixed scenarios. As detailed in Table 3, in the unsolvable scenario ($m = (0, 100)$) at graph size $n = 50$, SOFAI-v2 requires approximately 14,000 seconds, whereas both SOFAI_v1 and S2 take around 20,000 seconds. This corresponds to a substantial 30% reduction in average solving time. Similarly, for mixed solvability scenarios ($m = (50, 50)$) at the same graph size ($n = 50$), SOFAI-v2 achieves the same notable reduction of 30%, requiring only about 14,000 seconds compared to 20,000 seconds for SOFAI_v1 and S2. In fully solvable scenarios ($m = (100, 0)$), the improvements remain consistent. For instance, at graph size 50, SOFAI-v2 completes solving tasks in roughly 18,000 seconds, a 10% time reduction compared to SOFAI_v1 and S2 (each requiring about 20,000 seconds).

These time-efficiency gains are attributed to SOFAI-v2's advanced metacognitive governance, which intelligently manages solver interactions and computational resources based on real-time feedback and memory-driven decision-making. This dynamic resource allocation enhances solver efficiency, especially in challenging problem settings.

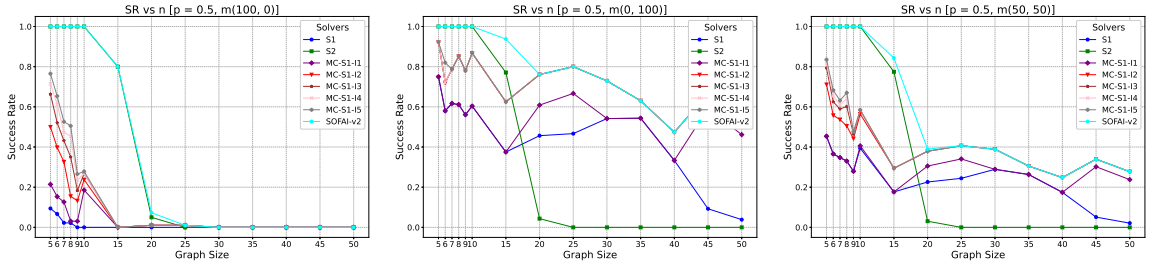
To ensure a fair and meaningful comparison of solver speed, we conducted this experiment without imposing any time limit. This is appropriate because SOFAI-v1, SOFAI-v2, and S2 are all complete solvers—that is, they can eventually solve or correctly determine the unsolvability of a problem given enough time. In contrast, S1 alone does not satisfy this property, so it is excluded from this timing comparison. This no-time-limit evaluation allows us to isolate and understand the time-efficiency advantages introduced by metacognitive governance in SOFAI-v2.

RQ 3

Does increasing the number of iterations in the feedback loop lead to an increased success rate in SOFAI-v2?

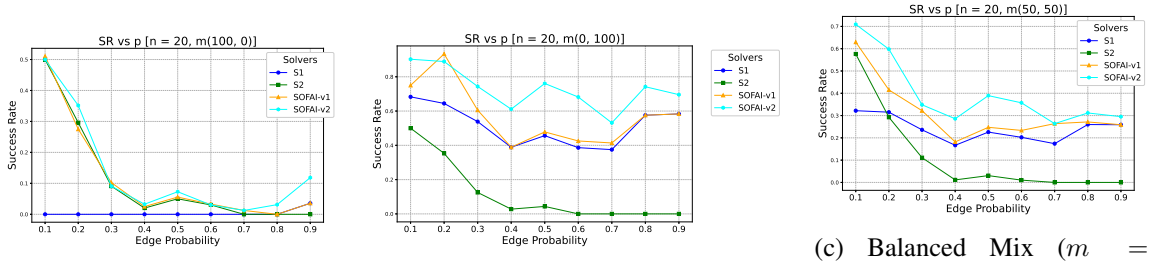
Answer to RQ 3: Yes, the iterative calling mechanism substantially enhances the performance of SOFAI-v2's S1 solver, driven by effective metacognitive governance. As illustrated in Figure 4, repeated solver invocations with structured feedback consistently improve success rates across all problem configurations:

- In fully solvable problems ($m = (100, 0)$), the initial success rate at iteration MC-S1-I1 (around 20% for graph size $n = 5$) improves substantially to nearly 80% by iteration MC-S1-I5, a notable increase of 300%.
- For completely unsolvable instances ($m = (0, 100)$) at $n = 10$, success rates begin at approximately 60% (MC-S1-I1) and increase to 85% by MC-S1-I5, an improvement of 40%.
- In balanced solvability scenarios ($m = (50, 50)$) at graph size $n = 10$, success rates also see meaningful improvements, increasing from approximately 40% (MC-S1-I1) to 60% (MC-S1-I5), representing a 50% improvement.



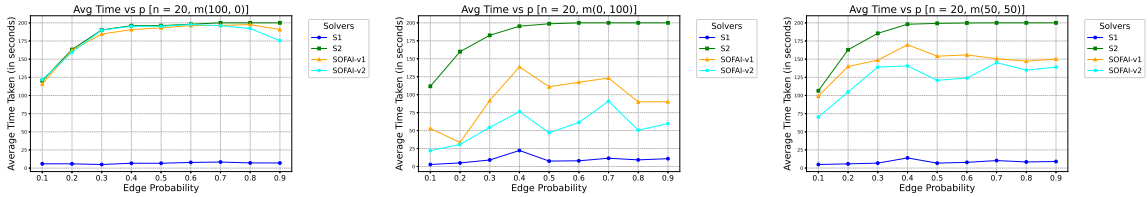
(a) All Solvable ($m = (100, 0)$) (b) All Unsolvable ($m = (0, 100)$) (c) Balanced Mix ($m = (50, 50)$)

Figure 4: Success rates of SOFAI-v2's S1 with iterative metacognitive feedback (MC-S1-I1 to MC-S1-I5) across different graph sizes and problem configurations, showing the impact of iterations on solver performance.



(a) All Solvable ($m = (100, 0)$) (b) All Unsolvable ($m = (0, 100)$) (c) Balanced Mix ($m = (50, 50)$)

Figure 5: Success rates across solvers as a function of edge probability in different problem configurations, highlighting the adaptability of SOFAI-v2 compared to traditional solvers.



(a) All Solvable ($m = (100, 0)$) (b) All Unsolvable ($m = (0, 100)$) (c) Balanced Mix ($m = (50, 50)$)

Figure 6: Time efficiency across solvers as a function of edge probability in different problem configurations, showcasing SOFAI-v2's performance in reducing average time taken.

These consistent improvements underscore how SOFAI-v2's metacognitive governance improves solver responses through iterative feedback, enhancing overall adaptability and efficiency across diverse problem settings. For the above results, we set a fixed time limit of 200 seconds per problem. If a solver does not finish within this time, it is counted as a failure.

RQ4

How does the density of graph influence success rate and time across the considered solvers?

Answer to RQ 4: Edge probability (p) substantially affects both solvability and solving time across graph coloring solvers, with SOFAI-v2 demonstrating superior adaptability and efficiency. Figure 5 shows that increasing p generally reduces success rates due to higher complexity from additional edges. However, SOFAI-v2 consistently maintains higher success rates compared to other solvers:

- In fully solvable problems ($m = (100, 0)$), all solvers maintain high success rates as p increases, with SOFAI-v2 performing comparably to S2 and SOFAI-v1.
- In completely unsolvable scenarios ($m = (0, 100)$), SOFAI-v2 notably outperforms others, particularly at lower edge probabilities. For instance, at $p = 0.1$, SOFAI-v2 achieves a 94% success rate, considerably surpassing SOFAI-v1 (74%), S1 (68%), and S2 (51%). This advantage remains pronounced (10-20%) even as p increases.
- In mixed scenarios ($m = (50, 50)$), SOFAI-v2 again excels, achieving a peak success rate of about 72% at $p = 0.1$, approximately 15-25% higher than other solvers.

Regarding time efficiency (Figure 6), as p increases, solving times rise initially and then stabilize. However, SOFAI-v2 consistently requires less time across all problem configurations:

- In the unsolvable scenario ($m = (0, 100)$), SOFAI-v2 solves problems substantially faster at lower and mid-range probabilities. For instance, at $p = 0.1$, SOFAI-v2 averages 20 seconds per problem, far less than SOFAI-v1 (55 seconds) and S2 (112 seconds).
- For mixed solvability scenarios ($m = (50, 50)$), SOFAI-v2 again demonstrates superior efficiency, 40-50% faster. Specifically, at $p = 0.1$, SOFAI-v2 solves problems in about 70 seconds, compared to SOFAI-v1's 100 seconds and S2's 108 seconds.

Overall, SOFAI-v2's robust metacognitive governance enables it to adapt efficiently and effectively to varying complexities induced by changing edge probabilities, achieving higher success rates and superior time efficiency. For the above results we set a fixed time limit of 200 seconds per problem. If a solver does not finish within this time, it is counted as a failure.

5.5 Other Considerations

We also explicitly address concerns regarding context window limitations inherent in modern LLMs. Although iterative prompting in our method increases token consumption, our analyses confirm that token usage remains comfortably within the Mistral-7B model's 32k-token context limit, even under extreme conditions (graph size 100, edge probability 0.9, and five iterations):

- Graph Representation (DIMACS format): approximately 1500 tokens
- Initial Prompt & Instructions: approximately 500 tokens

- Metacognitive Feedback per iteration (×5): approximately 2500 tokens (500 tokens each)
- Episodic Memory Retrieval per iteration (×5): approximately 5000 tokens (1000 tokens each)
- LLM Response per iteration (×5): approximately 5000 tokens (1000 tokens each)
- Accumulated Prompt per iteration (including previous responses, ×5): approximately 15,000 tokens (3000 tokens each iteration)

In total, even the worst-case scenario results in approximately 24,500 tokens, well within Mistral-7B's 32k-token context window. Consequently, context-window overflow is not an issue in our experiments. *Operationally, the metacognitive governance budgets information (problem, feedback, episodic examples) so that fast/automatic proposals remain verifiable within resource bounds, aligning with supervised control over routine processing (Norman & Shallice, 1986; Cooper & Shallice, 2000).*

Additionally, to account for the inherent variability in LLM output across runs, we repeat all experiments involving SOFAI-v1, SOFAI-v2, and S1 three times using the same set of problem instances. For success rate evaluation, a problem is considered a success if the solver succeeds in *any* of the three runs. For average time taken, we report the *minimum* time observed across the three trials. This choice reflects a conservative estimation of each system's upper potential in ideal conditions and avoids penalizing performance due to stochastic failures. Such a best-of-n (here n=3) strategy is commonly used in LLM evaluations to mitigate randomness and highlight consistent solver capabilities (Kang et al., 2025; Chow et al., 2024). *Cognitively, this also controls for noise in routine responding while assessing whether supervisory interventions reliably reduce errors.*

We also note that SOFAI-based solvers (and S1) can declare UNSAT without a constructive certificate, whereas S2's UNSAT arises from explicit search. This asymmetry can bias results on mixes containing unsatisfiable instances. However, the systematic variation in success rates across graph sizes and densities indicates the behavior is not arbitrary and reflects nontrivial problem sensitivity. A full proof-producing pipeline for UNSAT is orthogonal to our current scope; for automatic and quick evaluation, we accept such declarations, and we flag this as a limitation to be addressed in future work

5.6 Implications and Comparisons with Prior Work

This study broadens the scope of neurosymbolic systems by applying them to graph coloring problems. It demonstrates that integrating LLMs with traditional symbolic solvers, guided by metacognitive governance, enhances accuracy and efficiency. Unlike previous applications that primarily leveraged LLMs for sequential decision-making tasks (Valmeekam et al., 2022; Stechly et al., 2024), SOFAI-v2 directly addresses constraint-adherence limitations through iterative feedback and strategic use of symbolic reasoning. *Viewed through cognitive control, S1 provides routine proposals, while MC and S2 supply supervisory/controlled processing when conflicts persist, consistent with contention scheduling and supervisory attention (Norman & Shallice, 1986; Cooper & Shallice, 2000).* This neurosymbolic framing clarifies why SOFAI-v2 improves success rate and time efficiency on constraint satisfaction problems.

6. Conclusion and Future Work

We introduced SOFAI-v2, a neurosymbolic architecture that integrates a fast, episodic memory-based LLM (S1) with a slow, deliberative symbolic solver (S2), enhanced by real-time metacognitive governance. Our empirical evaluations across various graph coloring problems confirm that SOFAI-v2 outperforms traditional symbolic solvers and its predecessor, SOFAI-v1, in both success rates and time efficiency, especially within complex problem configurations. SOFAI-v2 employs iterative metacognitive feedback and episodic memory-based iterations to enable S1 to refine its response. This effectively overcomes the limitations of LLMs in constraint *adherence* for CSPs. By leveraging episodic memory and adaptive feedback, SOFAI-v2 improves the accuracy and efficiency of solving CSPs, indicating the value of cognitively informed, neurosymbolic architectures.

The observed improvements are consistent with dual-process theories: S1 generates a *fast, potential solution*; the metacognitive controller checks it for constraint adherence and uncertainty, requests targeted revisions when needed, and escalates to S2 only when necessary; S2 then performs *slow*, symbolic search as invoked. Episodic retrieval provides additional context from prior instances to support S1's proposal and the controller's checks. In this arrangement, SOFAI-v2 realizes a control loop that finds a trade-off between speed and reliability in a resource-constrained setting.

Future work will scale SOFAI-v2 to larger and more varied problem sets, explore different graph models, and apply the architecture to other CSPs and sequential decision-making tasks. We will also investigate learning-based MC policies and richer memory retrieval, and examine correspondences to cognitive-control signals (e.g., conflict costs) that could enable tighter links to computational cognitive neuroscience. *Concretely, we will design human-aligned probes (e.g., speed–accuracy manipulations, conflict-cost perturbations, and memory cueing/interference) to test whether SOFAI-v2's control dynamics mirror classic findings on supervisory attention and conflict-driven effort allocation.*

7. Acknowledgements

We thank Keerthiram Murugesan, Erik Miehling, Murray Campbell, Karthikeyan Natesan Ramamurthy, and Lior Horesh from IBM Research for their collaboration, valuable feedback, and overall support throughout the course of this research.

References

Bitner, J. R., & Reingold, E. M. (1975). Backtrack programming techniques. *Communications of the ACM*, *18*, 651–656.

Booch, G., et al. (2021). Thinking fast and slow in ai. *Proceedings of the AAAI Conference on Artificial Intelligence* (pp. 15042–15046).

Brélaz, D. (1979). New methods to color the vertices of a graph. *Communications of the ACM*, *22*, 251–256.

Brown, T. B. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Chaitin, G. J. (1982). Register allocation & spilling via graph coloring. *ACM Sigplan Notices*, *17*, 98–101.

Chow, Y., et al. (2024). Inference-aware fine-tuning for best-of-n sampling in large language models. *arXiv preprint arXiv:2412.15287*.

Cooper, R., & Shallice, T. (2000). Contention scheduling and the control of routine activities. *Cognitive Neuropsychology*, *17*, 297–338.

Dechter, R. (2003). *Constraint processing*. Morgan Kaufmann.

Fabiano, F., Ganapini, M. B., Loreggia, A., Mattei, N., Murugesan, K., Pallagani, V., Rossi, F., Srivastava, B., & Venable, K. B. (2025). Thinking fast and slow in human and machine intelligence. *Communications of the ACM*, *68*, 72–79.

Fabiano, F., Pallagani, V., Ganapini, M. B., Horesh, L., Loreggia, A., Murugesan, K., Rossi, F., & Srivastava, B. (2023). Plan-sofai: A neuro-symbolic planning architecture. *Neuro-Symbolic Learning and Reasoning in the era of Large Language Models*.

Ganapini, M. B., et al. (2022). Combining fast and slow thinking for human-like and efficient decisions in constrained environments. *NeSy* (pp. 171–185).

Golumbic, M. C. (2004). *Algorithmic graph theory and perfect graphs*. Elsevier.

Han, J., Collier, N., Buntine, W., & Shareghi, E. (2023). Pive: Prompting with iterative verification improving graph-based generative capability of llms. *arXiv preprint arXiv:2305.12392*.

Haralick, R. M., & Elliott, G. L. (1980). Increasing tree search efficiency for constraint satisfaction problems. *Artificial intelligence*, *14*, 263–313.

Jensen, T. R., & Toft, B. (2011). *Graph coloring problems*. John Wiley & Sons.

Jiang, Y., et al. (2023). Followbench: A multi-level fine-grained constraints following benchmark for large language models. *arXiv preprint arXiv:2310.20410*.

Johnson, D. S., & Trick, M. A. (1996). *Cliques, coloring, and satisfiability: second dimacs implementation challenge, october 11-13, 1993*, volume 26. American Mathematical Soc.

Kahneman, D. (2011). Thinking, fast and slow. *Farrar, Straus and Giroux*.

Kambhampati, S. (2024). Can large language models reason and plan? *Annals of the New York Academy of Sciences*, *1534*, 15–18.

Kang, Z., Zhao, X., & Song, D. (2025). Scalable best-of-n selection for large language models via self-certainty. *arXiv preprint arXiv:2502.18581*.

Khalil, E., Dai, H., Zhang, Y., Dilkina, B., & Song, L. (2017). Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, *30*.

Khandelwal, V., Pallagani, V., Srivastava, B., & Rossi, F. (2024). A neurosymbolic fast and slow architecture for graph coloring. *arXiv preprint arXiv:2412.01752*.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2022). Large language models are zero-shot reasoners. *Advances in neural information processing systems*, *35*, 22199–22213.

Kumar, V. (1992). Algorithms for constraint-satisfaction problems: A survey. *AI Magazine*, *13*, 32. From https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/976.

Li, Z., Chen, Q., & Koltun, V. (2018). Combinatorial optimization with graph convolutional networks and guided tree search. *Advances in neural information processing systems*, *31*.

Lin, B. Y., Fu, Y., Yang, K., Brahman, F., Huang, S., Bhagavatula, C., Ammanabrolu, P., Choi, Y., & Ren, X. (2024). Swiftsage: A generative agent with fast and slow thinking for complex interactive tasks. *Advances in Neural Information Processing Systems*, *36*.

Mackworth, A. K. (1977). Consistency in networks of relations. *Artificial intelligence*, *8*, 99–118.

Norman, D. A., & Shallice, T. (1986). Attention to action: Willed and automatic control of behaviour. In R. Davidson, G. Schwartz, & D. Shapiro (Eds.), *Consciousness and self regulation: Advances in research and theory, vol. 4*, 1–18. New York: Plenum.

Nye, M., et al. (2021). Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

Rossi, F., Van Beek, P., & Walsh, T. (2006). *Handbook of constraint programming*. Elsevier.

Ruoss, A., Delétang, G., Medapati, S., Grau-Moya, J., Wenliang, L. K., Catt, E., Reid, J., & Genewein, T. (2024). Grandmaster-level chess without search. *arXiv preprint arXiv:2402.04494*.

Schidler, A., & Szeider, S. (2023). Sat-boosted tabu search for coloring massive graphs. *ACM Journal of Experimental Algorithmics*, *28*, 1–19.

Stechly, K., Valmeekam, K., & Kambhampati, S. (2024). On the self-verification limitations of large language models on reasoning and planning tasks. *arXiv preprint arXiv:2402.08115*.

Valmeekam, K., Olmo, A., Sreedharan, S., & Kambhampati, S. (2022). Large language models still can't plan (a benchmark for llms on planning and reasoning about change). *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Verfaillie, G., & Jussien, N. (2005). Constraint solving in uncertain and dynamic environments: A survey. *Constraints*, *10*, 253–281. From https://doi.org/10.1007/s10601-005-2239-9.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., & Zhou, D. (2022). Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, *35*, 24824–24837.

Welsh, D. J., & Powell, M. B. (1967). An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, *10*, 85–86.

Yekezare, N., Zohrehbandian, M., Maghasedi, M., & Bonomo-Braberman, F. (2024). Optimality of dsatur algorithm on chordal graphs. *Operations Research Letters*, *57*, 107185.