Boundary–Programmable Prompting: Modular Design Patterns for Multi-Turn LLM Agents

Anonymous ACL submission

Abstract

Large language model agents are often governed by single-block prompts that entangle goals, heuristics, and memory, making their behavior brittle, opaque, and hard to control. We introduce **Boundary–Programmable Prompting (BPP)**, a modular prompting architecture that separates (i) boundary prompts encoding role and interpretive constraints, (ii) programmable heuristics that use *fuzzy heuristic ranges* to modulate behavior symbolically, and (iii) a short-term memory schema for tracking interaction-level state. Unlike prompting methods that rely on natural language alone, BPP provides machine-readable ranges that LLMs use to select strategies dynamically during inference. This structure supports more interpretable adaptation and bounds the space of valid actions to reduce hallucination. In a Socratic tutoring testbed, ablating each BPP module leads to selective, interpretable failures (e.g., loss of coherence without memory, rigidity without programmable heuristics, role drift without boundaries), demonstrating that modular prompting can serve as an inference-time control architecture.

1 Introduction

Large language models (LLMs) increasingly serve as the backbone of interactive systems (e.g., tutors, planners, coaches, assistants) where behavior must unfold coherently across turns and adapt to evolving user goals. Yet such agents often behave inconsistently: they lose track of prior context, apply heuristics unpredictably, or drift from their intended role (Laban et al., 2025; McCoy et al., 2024). A growing body of work suggests these failures stem not only from model limitations but from how prompts are structured. Most current approaches entangle goals, strategies, and context into a single instruction block, making behavior brittle, opaque, and hard to adapt (Wang et al., 2023; Liu et al., 2023; Morris, 2024).

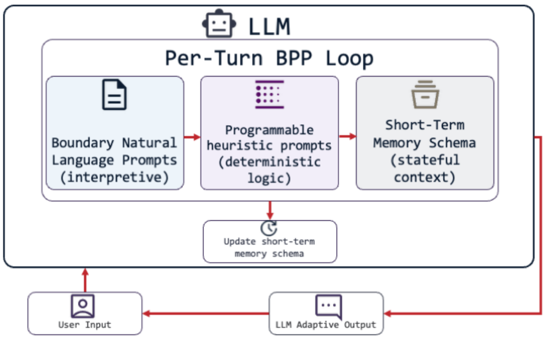

Figure 1: The per-turn Boundary–Programmable Prompting (BPP) loop. A single LLM operates over three modular components: a boundary prompt defining goals and interpretive rules, programmable heuristics providing decision logic based on numerical ranges the model reads and acts on at inference time, and a short-term memory schema maintaining state across turns. User input passes through this loop, the LLM generates an adaptive response, and memory is updated for the next turn.

Despite progress in prompting techniques, existing methods offer limited support for decomposing agent behavior into separable, independently controllable components. Techniques like few-shot prompting, chain-of-thought reasoning, and self-consistency decoding improve surface-level behavior but remain fundamentally monolithic (Brown et al., 2020; Wei et al., 2022; Zhou and Ai, 2024). Role framing and pedagogical prompting add structure (Andreas, 2022; Lee et al., 2024), but support only limited modularity (Bengio et al., 2009; Wang et al., 2024). Schema controllers guide input formats rather than decision logic (Okuda and Amarasinghe, 2024), and memory-augmented approaches retrieve static text rather than maintaining dynamic, interpretable state (Khandelwal et al., 2019; Lewis et al., 2020). Even in fuzzy control (Griol et al., 2021; Abduljabbar et al., 2023) and neural–symbolic prompting (d'Avila Garcez et al., 2019), inference-time modularization is rare.

Insights from multiple fields point toward a common architectural principle: adaptive behavior requires separable control components. Scaffolding theory distinguishes pedagogical intent from learner state (Vygotsky and Cole, 1978); fuzzy logic supports graded decision-making under uncertainty (Zadeh, 1996); and symbolic architectures emphasize explicit state and rule separation (Garcez et al., 2019). These perspectives converge on the need for modular controllers that structure interpretation and action selection.

We draw on these foundations to propose **Boundary–Programmable Prompting (BPP)**, a lightweight prompting architecture that decomposes control across three functional layers: (i) a *boundary prompt* that defines goals, role, and interpretive constraints; (ii) a *programmable heuristic layer* encoding symbolic or fuzzy decision logic; and (iii) a *short-term memory schema* that externalizes turn-level interaction state.

Crucially, BPP introduces machine-readable *fuzzy ranges*—graded control variables such as "confidence = low" or "support level = moderate"—that the LLM interprets to guide decisions at inference time. Rather than relying solely on implicit pattern recognition, the model reads explicit symbolic cues that modulate behavior. These fuzzy ranges bound the valid action space, improving adaptivity and reducing hallucination or drift.

BPP is not a new learning algorithm. Its contribution lies in prompt-level modularization: exposing symbolic surfaces for role framing, adaptive strategy, and memory—components that are often latent or interwoven in standard prompting. This structure improves control, interpretability, and reuse.

We validate BPP in a Socratic tutoring testbed [1]. In ablation experiments, removing each module yields targeted failures: coherence collapses without memory, adaptivity fails without fuzzy heuristic ranges, and role alignment breaks without boundaries. These interpretable patterns support the claim that symbolic modularity improves inference-time control.

Overall, BPP reframes prompting as architectural design. By organizing prompts into symbolic modules—and using fuzzy scores as explicit control signals—it provides a foundation for more robust, transparent, and adaptive LLM agents (Holtzman and Tan, 2025; Shah, 2025).

2 Boundary–Programmable Prompting (BPP)

BPP separates prompting into three functional modules—boundary prompts, programmable heuristics, and short-term memory—that together support adaptive, interpretable behavior. We illustrate BPP in an intelligent tutoring setting, but the framework is domain-general: any multi-step agent with alternative valid solution paths (e.g., games, task assistants, planning agents) can benefit from separating boundaries, decision rules, and state into explicit components.

2.1 Boundary Prompts: Declarative Constraints and Interpretation Rules

The boundary prompt specifies the agent's role, goals, and behavioral constraints, and critically defines how the remaining components should be interpreted. It instructs the model how to read state variables, sequence decision steps, and handle uncertainty. Prior work shows that role framing improves coherence (Andreas, 2022). BPP extends this idea by making the boundary prompt an explicit *interpretive layer* governing how rules and memory should be executed.

In practice, the boundary prompt answers: (1) *What agent are you?* (role, tone, goals) (2) *Which control schema applies?* (how to interpret heuristic ranges and rule conditions) (3) *How should memory be updated and used?*

This allows global behavior to be revised without altering heuristics or memory and provides a stable interpretive context across turns.

2.2 Programmable Heuristic Prompts: Structured Logic for Adaptive Behavior

Programmable heuristic prompts form the procedural layer, which is a structured set of symbolic and fuzzy heuristic ranges mapping user signals to actions. Fuzzy reasoning allows decision policies to be expressed over explicit, bounded heuristic ranges rather than inferred implicitly from raw dialogue, constraining how the model selects strategies under uncertainty. By conditioning action selection on interpretable ranges (e.g., confusion, knowledge level), the model operates within a predefined control envelope, reducing uncontrolled generalization and hallucination-prone behavior.

A key property of this layer is a *deterministic mapping* from state variables to actions. Unlike few-shot prompting or chain-of-thought methods

---

[1] Data available at: https://github.com/researcher2026/bpp

(Brown et al., 2020; Wei et al., 2022), which influence general reasoning tendencies, programmable heuristics define *exactly* which strategy applies in each region of the state space. This yields consistent behavior even under ambiguity and avoids drift toward generic or unstructured explanations.

2.3 Short-Term Memory Schema: Interpretable Stateful Context

The short-term memory schema stores an explicit, task-aligned record of the interaction. In our example, these cover misconceptions, prior scaffolding actions, learner progress, or affective cues. Retrieval-augmented systems typically pull external text (Lewis et al., 2020), whereas BPP maintains a lightweight symbolic store of *interaction-born state*. Memory updates follow simple rules in the boundary prompt, preventing regressions common in multi-turn dialogue and enabling adaptive adjustments.

A Micro-Example. If a learner says: *“I still don’t get why the Moon changes shape,”* then:

1. The **boundary prompt** classifies this as a clarification request and consults heuristic variables.
2. The **programmable heuristic schema** maps "high confusion" + "conceptual task" to a strategy: pose a guided question using a simplified analogy.
3. The **short-term memory schema** stores a misconception (e.g., "moon phases caused by Earth’s shadow") and increases the scaffolding level for the next turn.

This yields structured, interpretable behavior rather than ad hoc responses shaped by a monolithic prompt.

2.4 Modular Interaction and Why It Matters

Figure 1 illustrates the per-turn BPP loop, showing how boundaries constrain interpretation, heuristics govern decision-making, and memory enforces temporal coherence.

Interpretability. Each module is explicit and separable, allowing developers to localize failures to a boundary instruction, rule, or memory update rather than to an opaque prompt block.

Safety and Generality. Programmable heuristics act as auditable "control knobs" independent of model weights. Related work in pedagogical prompting and structured inputs (Lee et al., 2024; Okuda and Amarasinghe, 2024) hints at such structure, but BPP makes these components explicit.

3 Experimental Evidence for BPP

Our goal is not to achieve state-of-the-art tutoring performance, but to provide *mechanistic evidence* that the three BPP components(boundary prompts, programmable heuristics, and short-term memory) function as separable control mechanisms. We treat each module as a distinct control mechanism and ask: *what specific behavior degrades when it is removed?*

3.1 Setup

We evaluate BPP in two science domains (*global warming*, grade 7; *moon phases*, grade 11). For each domain, we construct five scenarios seeded with a simulated "uncertain learner." Each run yields a 5–7 turn dialogue, producing 50 dialogues total using LLaMA 3.1–8B.

To isolate architectural contributions, we compare five variants:

- **C0 (Full BPP):** boundary + heuristic ranges + memory
- **C1 (No-Memory):** removes state updates
- **C2 (No-Programmable-Heuristics):** removes graded heuristic ranges; static rules only
- **C3 (No-Boundary):** removes role framing and interpretation rules
- **C4 (Vanilla):** a standard single-block instructional prompt

We score model behavior along five tutoring-relevant dimensions: *scaffolding quality*, *contextual responsiveness*, *helpfulness*, *symbolic strategy use*, and *memory coherence*. An LLM rater applies a structured rubric (Appendix).

3.2 Results

Table 1 shows the quantitative pattern. The ablations yield a clean lesion profile: removing each module selectively impairs the behavior it is responsible for enabling.

Table 1: Mean evaluation scores across five tutoring dimensions by prompt condition.

Cond.	Scaf.	Resp.	Help.	Symb.	Mem.
C0 (Full)	4.80	4.88	4.76	4.72	4.64
C1 (No-Memory)	4.28	4.44	4.36	4.32	3.76
C2 (No-Programmable-Heuristics)	4.24	4.40	4.28	4.00	3.92
C3 (No-Boundary)	4.20	4.08	4.04	3.72	3.80
C4 (Vanilla)	3.80	3.72	3.60	3.24	3.00

- **C1 (No-Memory) → Temporal coherence collapses.** The model forgets prior turns, reintroduces resolved concepts, and contradicts earlier guidance. Other dimensions remain relatively intact, indicating that state, not reasoning heuristics, is what fails.

- **C2 (No-Programmable-Heuristics) → Adaptive behavior collapses.** Without programmable graded heuristics, tutoring becomes rigid and occasionally inconsistent, as the model must infer control state implicitly from context, leading to unstable strategy selection.

- **C3 (No-Boundary) → Role alignment collapses.** The model drifts into direct answering or encyclopedic exposition, ignoring Socratic norms. Other mechanisms still function, but without an interpretive frame they are inconsistently deployed.

- **C4 (Vanilla) → All control mechanisms collapse.** Topic drift, repetitive explanations, inconsistent scaffolding, and poor knowledge tracking match known failure modes of monolithic prompting.

These lesion outcomes match BPP's intended decomposition: memory controls continuity, heuristics control adaptivity, boundaries control role and framing. Even a minimal implementation produces clean, interpretable failure modes, supporting modular prompting as a plausible design pattern for multi-turn LLM control.

4 Conclusion

This paper proposed **Boundary-Programmable Prompting (BPP)**, a modular view of prompting that separates declarative boundaries, programmable heuristics, and short-term memory into distinct components of control. Rather than treating prompting as a monolithic specification, BPP frames LLM behavior as the coordinated interaction of these layers: boundaries define global intent and interpretation rules, heuristics supply adaptive decision policies, and memory provides continuity across turns.

Our early implementation provides preliminary support in that ablating each module selectively degrades the behavior it is meant to control, supporting the view that modular prompting is a strong architectural *pattern* for disentangling control surfaces in LLM agents. Many existing LLM systems (e.g., planners, multi-step solvers, coaching agents, retrieval-augmented assistants) implicitly rely on similar functions; BPP makes these elements explicit and inspectable.

By modularizing prompt components, developers can revise and debug agent behavior systematically, reducing brittleness and improving reliability. As LLMs transition from text generators to interactive agents, architectural clarity becomes increasingly important. BPP offers a concise conceptual foundation for designing adaptive, transparent prompting systems and provides a starting point for future work on richer neurosymbolic prompt architectures.

Limitations

We present BPP as a conceptual framework supported by preliminary evidence rather than a comprehensive empirical study. Our experiments are limited to synthetic learners, a single model family, and LLM-based evaluation; human studies and broader domains are needed to assess external validity. The framework assumes that LLMs reliably follow modular prompt components, an assumption that may fail under distribution shift or adversarial inputs. Additionally, designing programmable heuristics and memory states still requires manual effort and domain expertise, which may not scale to complex settings without tooling support. Future work should probe BPP across tasks, models, and users, and investigate methods for learning or refining heuristic and memory structures from data.

References

Abduljabbar Khudhur Abduljabbar, Yousif Al Mashhadany, and Sameer Algburi. 2023. High-performance of mobile robot behavior based on intelligent system. In *2023 16th International Conference on Developments in eSystems Engineering (DeSE)*, pages 445–450.

Jacob Andreas. 2022. Language models as agent models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5769–5779.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, number 8 in ICML '09, pages 41–48, New York, NY, USA. Association for Computing Machinery.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Artur d'Avila Garcez, Marco Gori, Luis C. Lamb, Luciano Serafini, Michael Spranger, and Son N. Tran. 2019. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. arXiv.

Artur d'Avila Garcez, Tarek R. Besold, and Luis C. Lamb. 2019. Neurosymbolic AI. *Artificial Intelligence*.

David Griol, Zoraida Callejas, Jose Manuel Molina, and Araceli Sanchis. 2021. Adaptive dialogue management using intent clustering and fuzzy rules. *Expert Systems*, 38(1):e12630.

Ari Holtzman and Chenhao Tan. 2025. Prompting as scientific inquiry. *arXiv preprint arXiv:2507.00163*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. 2025. Llms get lost in multi-turn conversation. *Preprint*, arXiv:2505.06120.

Unggi Lee, Jiyeong Bae, Yeonji Jung, Minji Kang, Gyuri Byun, Yeonseo Lee, Dohee Kim, Sookbun Lee, Jaekwon Park, Taekyung Ahn, and 1 others. 2024. From prediction to application: Language model-based code knowledge tracing with domain adaptive pre-training and automatic feedback system with pedagogical prompting for comprehensive programming education. Arxiv.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

R. Thomas McCoy, Shunyu Yao, Dan Friedman, Mathew D. Hardy, and Thomas L. Griffiths. 2024. Embers of autoregression show how large language models are shaped by the problem they are trained to solve. *Proceedings of the National Academy of Sciences*, 121(41):e2322420121.

Meredith Ringel Morris. 2024. Prompting considered harmful. *Communications of the ACM*, 67(12):28–30.

Katsumi Okuda and Saman Amarasinghe. 2024. Askit: Unified programming interface for programming with large language models. In *Proceedings of the 2024 IEEE/ACM International Symposium on Code Generation and Optimization*, CGO '24, pages 41–54. IEEE Press.

Chirag Shah. 2025. From prompt engineering to prompt science with humans in the loop. *Communications of the ACM*, 68(6):54–61.

L. S. Vygotsky and Michael Cole. 1978. *Mind in society: the development of higher psychological processes*. Harvard University Press, Cambridge.

Xin Wang, Yuwei Zhou, Hong Chen, and Wenwu Zhu. 2024. Curriculum learning: Theories, approaches, applications, tools, and future directions in the era of large language models. In *Companion Proceedings of the ACM Web Conference 2024*, WWW '24, pages 1306–1310, New York, NY, USA. Association for Computing Machinery.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of ACL*, pages 13484–13508.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Lotfi A. Zadeh. 1996. Fuzzy logic = computing with words. *IEEE Transactions on Fuzzy Systems*, 4(2):103–111.

Yuhang Zhou and Wei Ai. 2024. Teaching-assistant-in-the-loop: Improving knowledge distillation from imperfect teacher models in low-budget scenarios. arXiv.

A Prompting Framework Details

A.1 Symbolic Prompt Architecture

BPP integrates three symbolic control layers into the prompting pipeline: (1) a boundary prompt defining role and interpretive rules, (2) a programmable heuristics encoding graded decision policies, and (3) a short-term symbolic memory that maintains state across turns. Together, these components implement a lightweight control loop without modifying model weights.

A.2 Boundary Prompt Structure

The boundary prompt establishes the agent role, global goals, and instructions for interpreting the programmable heuristic and memory schemas. This is described in the example (excerpt) below.

```
You are an adaptive tutor. Interpret the control
     schema as follows:
1. Identify task type using `task_types`.
2. Select a scaffolding strategy based on `
    knowledge_level`.
3. Adjust complexity using `readability_levels`.
4. Update memory after each turn with new
    misconceptions,
   progress indicators, or affective cues.
```

This layer specifies how the model should *read* the schema and when to update state, functioning as the interpretive “meta-control” component.

A.3 Programmable Heuristic Schema

The programmable heuristic schema encodes graded instructional logic using fuzzy sets and simple rules. The example (excerpt) below illustrates this.

```
{
  "knowledge_levels": {
    "emerging": 1,
    "developing": 2
  },
  "scaffolding_types": {
    "high": 3,
    "moderate": 2
  },
  "mappings": {
    "emerging": "high",
    "developing": "moderate"
  }
}
```

The schema maps learner signals (e.g., confusion, progress, confidence) to graded strategies, providing an interpretable and deterministic decision ladder.

A.4 Short-Term Memory Schema

Memory captures interaction-born state rather than generic history, enabling coherence across turns. The following excerpt exemplifies this.

```
{
  "task_topic": "moon phases",
  "knowledge_level": 1,
  "misconceptions": ["confuses shadow with tilt
    "],
  "scaffolding_history": ["high", "moderate"]
}
```

Updates follow rules defined at the boundary layer (e.g., append new misconceptions, escalate scaffolding after repeated help requests).

Table 2: Ablation conditions and observed effects.

Cond.	**Component removed**	**Summary of effect**
C0	None	Full BPP; strongest performance across all metrics.
C1	Memory schema	Large drop in *Memory*; minor declines in scaffolding and symbolic strategy use.
C2	Programmable heuristics	Loss of graded adaptivity; weaker scaffolding and responsiveness.
C3	Boundary prompt	Role drift; broad degradation, especially in helpfulness and responsiveness.
C4	All symbolic layers	Generic zero-shot tutoring; lowest performance overall.

B Evaluation Rubric (Summary)

Each assistant response was rated on five dimensions: (1) scaffolding quality, (2) contextual responsiveness, (3) helpfulness, (4) symbolic strategy use, (5) memory coherence.

Scores ranged from 1–5 with rubric-based guidance. A full example rater prompt is provided in the project repository.

C Ablation Mapping and Component Impact

An overview of the ablation conditions and observed effects is shown in table 2.

D Results Summary

Scores follow the predicted modular pattern: removing each component selectively impairs the behavior it controls, supporting the functional decomposition proposed by BPP (Figure 2.

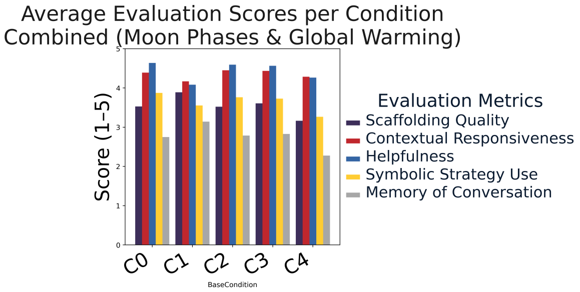

Figure 2: Average evaluation scores across experimental conditions.