COP: A Memory-Augmented Chain-of-Paradigms Framework for Inductive Reasoning

Anonymous ACL submission

Abstract

Large language models (LLMs) have achieved strong performance in text generation, yet their inductive reasoning processes often exhibit instability and limited generalization across tasks. In this work, we propose Chain of Paradigms (COP), a memory-augmented inductive reasoning framework that enables reusable high-level reasoning patterns to be stored, retrieved, and instantiated during inference. COP consists of a problem expander for extracting task-critical information, a lightweight paradigm buffer that maintains structured reasoning patterns, and a dynamic retrieval mechanism that selects relevant paradigms via semantic matching. These components form a closed-loop reasoning process that supports pattern reuse across tasks while mitigating erratic inference behaviors. We evaluate COP on the Big-Bench Hard (BBH) benchmark using exact match accuracy, inference cost, and cross-task pattern reuse metrics, with controlled comparisons against existing prompting and agent-based reasoning methods. Experimental results demonstrate consistent improvements in accuracy and robustness over strong baselines, while maintaining efficient inference. This work enhances the reliability of generative models in complex reasoning tasks, provides insights into aligning AI inference with human cognitive patterns, and contributes to interdisciplinary research at the intersection of cognitive psychology and AI alignment.

1 Introduction

Research Background and Significance. Generative models have shown strong capabilities in natural language processing (Bhatia, 2023), particularly in text generation, question-answering systems, and conversational agents. However, these models still have limited capabilities in inductive reasoning (Kambhampati, 2024). Inductive reasoning refers to the ability to derive general conclusions from a limited number of instances, which is essential for model performance in complex tasks. While Large Language models (LLMs) can predict logical relationships between sentences (e.g., entailment or contradiction) in natural language reasoning tasks (Sternberg, 1988), they fail to replicate human inductive reasoning mechanisms observed in psychological studies (Fierro Celis and Andrade Navia, 2022). This highlights that the enhanced "planning ability" of LLMs is essentially an optimization of approximate retrieval, rather than a true understanding of the problem's logic (García-Campos and Sarabia-López, 2022).

Research Problems and Objectives. Existing research has focused on improving the reasoning abilities of LLMs through cue engineering and multi-query reasoning methods, but these approaches lack generalization and scalability (Choi et al., 2023; Chen et al., 2024). For example, single-query reasoning approaches (e.g., CoT (Wei et al., 2022) and few-shot cueing (Wang et al., 2020)) requires task-specific cue design and lack generalizability. Multi-query reasoning methods (e.g., Least-to-Most (Zhou et al., 2023), Tree-of-Thoughts (Yao et al., 2023a), Graph-of-Thoughts (Besta et al., 2024a), Question Rephrasing (Li et al., 2024)) solve complex problems by decomposing them into sub-problems but require multiple calls to the model (Huang and Chang, 2022). This study aims to design a new Chain of Paradigms (COP) method to enhance the inductive reasoning capabilities of LLMs by introducing an inductive thinking paradigm. Specifically, we aim to construct a paradigm-buffer and instantiate these patterns to achieve efficient reasoning.

Theory and Motivational Foundations. Inductive reasoning plays a central role in human cognition (Xu et al., 2025), allowing us to distill general laws from limited examples. Integrating this capability into LLMs not only enhances the mod-

els' reasoning ability but also better adapts them to complex tasks. Furthermore, by constructing reusable thinking data patterns, we provide structured knowledge support for subsequent model generation tasks (Huang and Chang, 2023), promoting the deeper development of LLMs in language understanding and generation. Additionally, this research offers new perspectives and methods for interdisciplinary fields such as brain science and cognitive psychology (Rajani et al., 2019), with significant theoretical and practical implications. This study builds on prior research, addresses the reasoning limitations of existing models, and proposes a thinking paradigm to enhance the inductive reasoning capabilities of LLMs.

In summary, our contributions include: **First**, we incorporate inductive thinking into the generation process of generative models. By integrating the BBH (Big-Bench-Hard) benchmark dataset, we verify that inductive thinking enhances model generation results. **Second**, we designed a thinking data structure based on the inductive paradigm, which includes the elements of COTs (context), cases (examples), patterns, and verification, and further optimized the dataset quality via reliability assessment. **Third**, COP is driven by Memory-Augmented and can automatically adapt the paradigm for different reasoning tasks, significantly improving the model's inference accuracy and generative power. **Fourth**, by constructing reusable thinking data maps and designing generative models for unknown thinking paradigms, this research provides new perspectives and methods for interdisciplinary fields such as brain science and cognitive psychology.

2 Related Work

Multi-Query Reasoning. Multi-query reasoning methods decompose complex problems by querying LLMs multiple times to generate various reasoning paths. Specifically, these methods include subproblem decomposition with iterative hinting, Tree of Thought (ToT) (Yao et al., 2023b) and path search, retrieval enhancement and knowledge graph fusion, as well as self-consistency and diversity optimization. For example, the iterative hinting technique in (Dua et al., 2022) allows subsequent subproblems to access antecedent results, while (Press et al., 2023) uses CoT hints to complete the decomposition in a single forward pass. The Tree-of-Thoughts (ToT) and Graph-of-Thoughts (GoT) (Besta et al., 2024b) methods improve model inference capability by constructing a tree or graphical inference structure. The RoG framework (Luo et al., 2023), SearChain (Jiang et al., 2020), and Readi (Cheng et al., 2024) frameworks improve model reasoning by fusing query reasoning with knowledge graphs. Generalization is improved by sampling multiple reasoning paths and voting for the most consistent answer (self-consistency) (Wang et al., 2023) or introducing diverse examples. However, these approaches are computationally expensive and rely on manually designed reasoning structures, which lack flexibility.

Inductive Thinking Paradigm Interpretation. Inductive thinking is a fundamental cognitive process that derives general principles or patterns from a finite set of observed instances (Binti Misrom et al., 2020). Unlike deductive reasoning, which applies predefined rules to specific cases, inductive reasoning infers latent regularities from empirical evidence and extends them to unseen situations. This process has been widely adopted in qualitative research and cognitive science, where general theories are constructed through the analysis and abstraction of concrete examples (Peltonen, 2022; Mott and Bullock, 2015). By encouraging higher-order reasoning and pattern abstraction (Riguzzi et al., 2014; Hammer, 2011), inductive thinking enhances problem-solving capabilities across diverse domains.

Formally, the inductive derivation process can be described as follows. Given a finite set of observations $S_1, S_2, \ldots, S_n \in S$, if each observation S_j exhibits a shared property or pattern P_{pat}, i.e.,

$$S_j \to P_{\text{pat}}, \quad j = 1, 2, \ldots, n, \tag{1}$$

then it is reasonable to generalize that the entire class S satisfies the same pattern, denoted as $S \to P_{\text{pat}}$. This classical form of inductive generalization can be abstracted into a hypothesisevidence framework:

$$P + O \to R, \tag{2}$$

where P represents a generalized pattern, O denotes the set of observed evidence, and R indicates that the hypothesis is accepted as reasonable under the given observations.

From a probabilistic perspective, inductive reasoning can be further formalized as a conditional in-

ference process. Under the assumptions of hypothesis expressibility, model recognizability, and sufficient data quality, inductive generalization corresponds to the convergence of posterior belief:

$$\lim_{|O|\to\infty} \Pr(P \mid O) = 1, \tag{3}$$

indicating that the hypothesis becomes almost surely valid as supporting evidence accumulates (Kirkegaard, 2009).

In practical computational systems, however, exhaustive observation is infeasible. Therefore, inductive reasoning is implemented as a finite-sample approximation, where a hypothesis is accepted only if it remains consistent under empirical verification. In this paper, we compress the above probabilistic induction process into an operational form:

$$P = \Phi(O), \quad O = \{O_1, \ldots, O_n\}. \tag{4}$$

where $\Phi(\cdot)$ denotes an abstraction operator that compresses finite observations into a reusable inductive pattern.

BBH Dataset. BIG-Bench dataset is a collaborative benchmark designed to quantitatively assess the strengths and weaknesses of language models (Srivastava et al., 2022). It includes over 200 diverse text-based tasks across categories such as traditional NLP, mathematics, commonsense reasoning, and question answering. The remaining **23 tasks** form our curated benchmark, BIG-Bench Hard (BBH), which includes two label: Logical Deduction and Tracking Shuffled Objects, each with three subtasks. For all tasks in BBH, except for three, we selected a random subset of 250 evaluation examples, totaling 6,511 examples in the benchmark.

3 Chain of Paradigm

This paper introduces a structured inductive reasoning framework grounded in inductive thinking theory (Section 3.1). To operationalize this framework, we design a set of tightly coupled modulesincluding a question expander (Section 3.2), a paradigm-buffer, pattern retrieval, embedded reasoning (Section 3.3), and a Paradigm Manager (Section 3.4) that jointly improve generative reasoning performance on the BBH benchmark. In addition, we provide a unified data structure for modeling inductive reasoning processes. Details of the COP framework implementation are provided in Appendix A.

As illustrated in Figure 1, the question expander extracts key information and generates candidate paradigms; the paradigm-buffer stores reusable high-level reasoning patterns for cross-task sharing; pattern retrieval identifies relevant paradigms via embedding similarity; embedded reasoning instantiates paradigms into task-specific reasoning paths; and the Paradigm Manager continuously updates the buffer to improve stability and generalization.

3.1 Inductive Paradigm Structuring

The COP data structure is designed as a computational abstraction of inductive reasoning, where generalization emerges from structured observations rather than being explicitly hard-coded. Instead of treating reasoning traces as unstructured text, COP decomposes inductive inference into six interacting components:

$$COP = Q, A, Co, Ca, P, R. \tag{5}$$

This formulation explicitly models the inductive pipeline in COP: structured observations are first extracted from the input question, generalized into an abstract pattern, and subsequently instantiated to generate the final answer.

$$\begin{aligned} &\Phi(Q) = (Co, Ca), \\ &P \sim \Pr(P \mid \Phi(Q)), \\ &A = Gen(Q, P), \\ &R = \mathbb{I}\left[\mathrm{Sim}(P, \Phi(Q)) \geq \tau\right]. \end{aligned} \tag{6}$$

Here, $\Phi(\cdot)$ denotes an observation extraction operator that converts an input question into structured reasoning traces and concrete cases. $\mathrm{Sim}(\cdot)$ measures semantic consistency using embedding-based similarity, and τ controls the minimal consistency required for a pattern to be considered valid.

Question (Q). Q represents the input problem drawn from the 23 BBH tasks.

Answer (A). A denotes the model-generated solution corresponding to Q.

COTs (Co). Co captures structured reasoning traces that analyze Q, providing intermediate interpretations and decompositions.

Cases (Ca). Ca consists of concrete instances or contextual conditions extracted from Co, serving as empirical evidence for pattern induction.

Patterns (P). P represents the abstract principle that captures common structures shared by

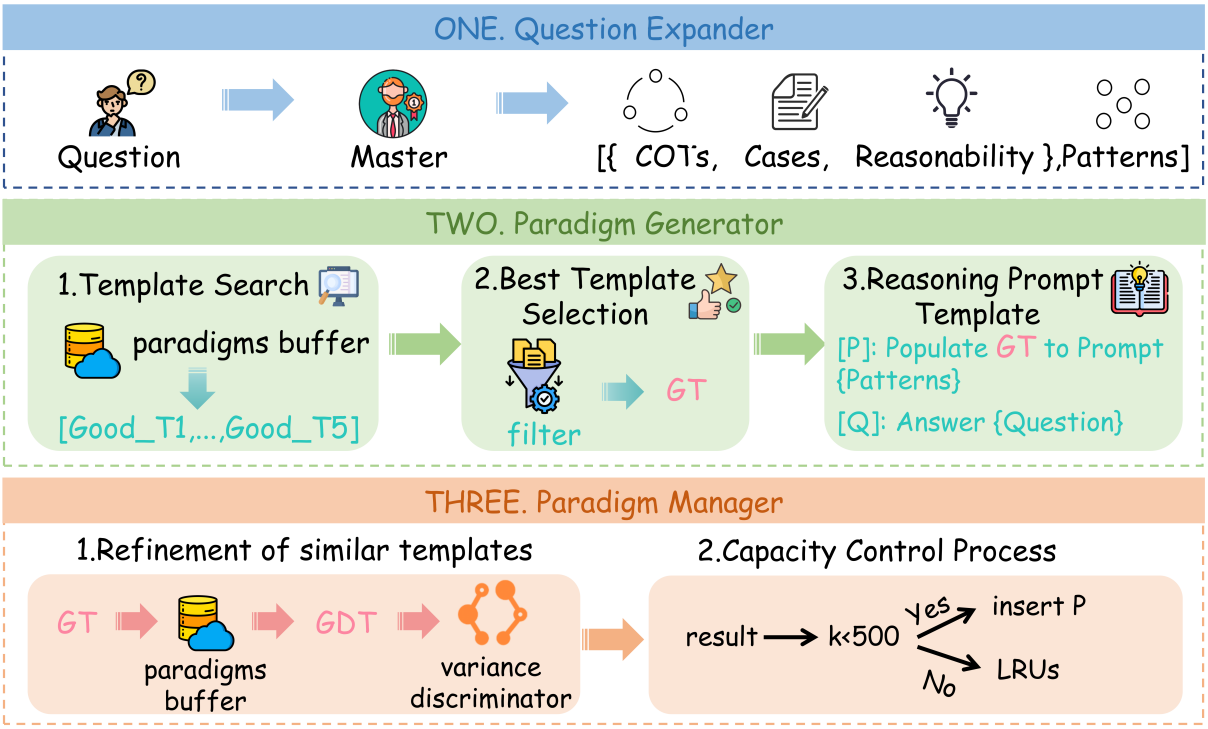

Figure 1: The research framework in this paper consists of the following core modules: the question expander (Section 3.2) extracts key information from the input and generates paradigms; the pattern Generator filters the most relevant patterns through embedded similarity computation, generating reasoning paths by combining the patterns with specific reasoning steps via embedded reasoning (Section 3.3); and the paradigm manager (Section 3.4) dynamically optimizes the bufferrefining new patterns to enhance system capability and using LRT to optimize the buffer.

(Co, Ca), enabling knowledge generalization and reuse across tasks.

Reasonability (R). R evaluates whether an induced pattern P remains logically consistent and explanatory when confronted with potential counterexamples.

From a formal perspective, COP models inductive reasoning as a conditional generalization process. Given observations composed of reasoning traces and cases (Co, Ca), the objective is to induce a reusable pattern P that maximizes explanatory consistency:

$$P^* = \arg\max_{P \in \mathcal{P}} \Pr(P \mid O), \tag{7}$$

where O denotes observed evidence and P denotes an inductively generalized pattern. This formulation explicitly connects COP to probabilistic inductive inference, where patterns are treated as hypotheses conditional on observed evidence.

By integrating inductive theory with structured reasoning representations, COP provides **a unified cognitivecomputational framework** for studying and implementing inductive reasoning in generative models.

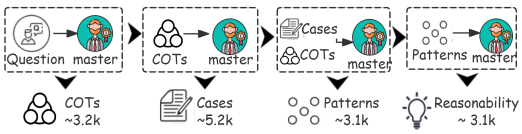

Figure 2: Inductive Paradigms Structure. COP captures the core units of inductive question-answering logic by defining four key cognitive nodes.

3.2 Question Expander

The question expander serves as the inductive entry point of COP, aiming to extract structured observations from the input task and approximates inductive pattern inference. Specifically, it transforms raw questions into candidate paradigms by decomposing them into reasoning traces and cases.

Problem expansion. The distiller decomposes Q into a reasoning trace Co and a set of cases Ca, and then induces a candidate pattern $\hat{P}$:

$$\hat{P} \approx \text{LLM}\big(\Phi(Co, Ca)\big), \tag{8}$$

where $\psi(\cdot)$ is a paradigm extraction prompt. Here, the LLM is used as a finite-sample approximation of inductive inference rather than as a free-form generator.

Pattern abstraction. The extracted evidence (Co, Ca) and verification signals are summarized into a task descriptor, which is further abstracted into a candidate inductive paradigm $\hat{P}$ suitable for cross-task reuse.

Key information extraction. From the expanded results, we extract implicit information $E = \{Co, Ca, \text{verify}\}$, and paradigm features p, forming a task descriptor $\hat{P} = (E, p)$. This descriptor is further abstracted into a meta-pattern capable of generalizing across structurally similar tasks.

3.3 Paradigm Generator

Pattern retrieval. Given a candidate paradigm $\hat{P}$, relevant paradigms are retrieved from the buffer via embedding similarity:

$$GoodPs = \arg \max_{DT_i \in PB} \text{Sim}\big(f(\hat{P}), f(DT_i)\big), \tag{9}$$

where $PB = DT_i$ is the paradigm-buffer and $f(\cdot)$ is an embedding model.

$$P^* \approx \arg \max_{P \in GoodPs} \Pr(P \mid Co, Ca). \tag{10}$$

Embedded reasoning. The selected pattern P^* is instantiated by injecting task-specific values from (Co, Ca) into pattern placeholders, yielding a concrete reasoning path that guides final answer generation. The prompt construction template based on inductive reasoning mapping is provided in Appendix H.

3.4 Paradigm Manager

The paradigm buffer (PB) maintains a collection of high-level inductive reasoning patterns $\{DT_i\}_{i=1}^{N}$ distilled from historical tasks, where each paradigm encodes a reusable abstraction of structured reasoning processes rather than task-specific solutions. By accumulating paradigms across diverse tasks, PB serves as a long-term inductive memory that supports cross-task generalization and mitigates the brittleness of single-instance reasoning.

The paradigm manager maintains the quality and efficiency of the paradigm buffer through two complementary principles: **inductive reasonability** and **capacity control**. Inductive reasonability filters newly induced paradigms by enforcing logical consistency with observed reasoning traces and cases, preventing the accumulation of spurious or weakly grounded patterns. Capacity control constrains buffer growth to balance generalization and computational efficiency, ensuring stable retrieval and transferable inductive knowledge over time.

Pattern refinement. Given a newly induced pattern GT, its compatibility with each existing paradigm $DT_i \in PB$ is evaluated using a similarity-based score:

$$\gamma_i = \text{Sim}(GT, DT_i). \tag{11}$$

Let $\gamma^* = \max_i \gamma_i$ denote the highest compatibility score. If $\gamma^* < \tau$, the pattern GT is considered insufficiently explained by existing paradigms and is therefore admitted as a new inductive pattern into the paradigm buffer.

Otherwise, the most compatible paradigm is selected as:

$$GDT = \arg \max_{DT_i \in PB} \gamma_i. \tag{12}$$

Although the selection rule is capacity-agnostic, the emergent inductive behavior of the paradigm buffer depends on both the update strategy and the buffer capacity N. Limited capacity biases the system toward recent patterns, favoring short-term adaptation at the expense of long-range abstraction. Increasing N mitigates this bias by preserving low-frequency but high-transfer paradigms, enabling cross-task generalization.

Capacity control. To balance generalization and efficiency, a Least Recently Used (LRU) policy is adopted to maintain the buffer within a fixed capacity. In all experiments, the buffer size is set to $N = 500$.

4 Experiments

4.1 Experimental Setup and Evaluation

Experimental Objectives. [1] The experiments aim to verify whether the proposed COP framework, which explicitly models inductive reasoning through reusable cognitive paradigms, can: improve reasoning accuracy across heterogeneous tasks, enable effective cross-task pattern reuse, and maintain acceptable inference efficiency under limited computational resources.

Baselines. We compare COP against three categories of baselines: **(i) Multi-query reasoning methods**, including Zero-shot, Chain-of-Thought

[1] Detailed implementation details and simulation examples are provided in the supplementary material.

(COT), Question Rephrasing (QReph), and Reverse Thinking (Reverse), and ReAct (Yao et al., 2023c), and Reflexion (Shinn et al., 2023), and Tree-of-Thoughts (TOT), and Graph-of-Thoughts (GOT). which enhance performance through intra-query reasoning expansion; **(ii) Vectorized representation baselines**, using jina-embeddings-v2, which rely on semantic similarity without explicit inductive abstraction; and **(iii) Large-scale generative baselines**, including Yi-34B-chat (Young et al., 2024), Llama-3.x series (Guo et al., 2024; Cook et al., 2024; Dong et al., 2024), and QWEN variants (Bai et al., 2023; Qwen et al., 2025), covering a wide range of model capacities.

Implementation Details. All experiments are implemented using the HuggingFace Transformers framework. We adopt jina-embeddings-v2 (Sturua et al., 2024) as the default encoder. The embedding dimension of all cognitive patterns is fixed at **768**, matching the encoder hidden size to ensure semantic alignment. A dynamic update strategy is applied to the paradigm-buffer: after every **100** tasks, low-frequency or low-confidence patterns are replaced following an incremental learning strategy inspired by BOT. To balance domain specialization and generalization, cross-task examples constitute 30% of the training data and are mixed with within-task examples at a 7:3 ratio. All experiments are conducted on two NVIDIA GTX 1080Ti GPUs (12GB). See Appendix C for detailed experimental parameters.

Evaluation Metrics. Performance is evaluated along three complementary dimensions: **Exact Matching (EM)**, measuring answer correctness; **Average Reasoning Time (ART)**, measuring end-to-end inference efficiency; and **Cross-task Template Reuse Rate (CTRR)**, quantifying the proportion of reasoning patterns reused across different tasks, which directly reflects inductive generalization ability.

4.2 Evaluation on Complex Reasoning

Analysis. Table 1 compares COP with prompting-based and agent-style reasoning methods across models of different scales. Two consistent trends can be observed. **First**, COP achieves the highest EM scores across almost all model sizes, particularly on small and medium models such as Llama-1B and Llama-3B. This suggests that **inductive pattern reuse effectively compensates** for limited parametric reasoning capacity. **Second**, while agent-based methods (e.g., Tree-of-Thoughts and Graph-of-Thoughts) also improve accuracy, they incur substantially higher inference costs. In contrast, COP attains comparable or better accuracy with significantly lower reasoning overhead, indicating a more favorable accuracy-efficiency trade-off. A comprehensive comparison with traditional CoT methods is presented in Appendix B.

4.3 Evaluation on Cross-task Generalization

Analysis. Table 2 jointly analyzes the effects of buffer management strategies and capacity scaling. Across all strategies, increasing the buffer capacity consistently improves CTRR, confirming that inductive generalization benefits from a richer pattern repository. However, larger capacities also introduce higher retrieval latency and memory overhead. Among the three strategies, the mixed phase-out approach achieves the best trade-off, reaching the highest EM at capacity $N = 200$ and the highest CTRR at $N = 500$. Dynamic LRU minimizes inference latency but exhibits weaker long-term inductive retention, particularly under small buffer settings. These results indicate that both buffer size and elimination policy are critical factors for cross-task inductive reasoning.

Analysis. The mixed phase-out strategy achieves the highest EM, indicating more effective long-term inductive knowledge retention. Dynamic LRU provides the lowest inference latency by aggressively discarding low-utility patterns, at the cost of higher memory usage. These results reveal a clear trade-off between efficiency and inductive coverage, motivating adaptive buffer management strategies in COP.

5 Ablation Study

5.1 Buffer Capacity and Update Strategy

Setup. We conduct an ablation study to analyze how paradigm-buffer capacity N interacts with different buffer management strategies. We evaluate three strategiesFixed Capacity, Dynamic LRU, and Mixed Phase-outunder three buffer sizes ($N \in \{50, 200, 500\}$). All other components are kept identical.

Impact of Buffer Capacity. As shown in Table 2, increasing buffer capacity consistently improves CTRR across all strategies, confirming that larger buffers enable stronger inductive generalization by retaining reusable paradigms. However, this gain comes with increased retrieval cost and

Table 1: Comparison of Accuracy (EM) and Inference Cost (ART, Seconds) Across Models and Reasoning/Agent-Based Methods: COP (Fixed Capacity, FC), COP (Dynamic LRU, DLRU), and COP (Mixed Phase-out, MPO). Blue Cells Indicate the Best Performance Per Model, While Gray Cells Denote the Lowest EM Baselines.

Method	Llama-1B		Llama-3B		Llama-8B		Yi-34B		QWEN Plus		QWEN Turbo	
	EM	ART	EM	ART	EM	ART	EM	ART	EM	ART	EM	ART
Zero-shot	32.33	**18.02**	26.06	**19.01**	47.62	**21.03**	21.11	**25.04**	35.67	**28.02**	22.10	**26.01**
CoT	35.65	30.18	30.04	32.21	40.48	35.17	24.57	38.26	58.24	40.33	28.29	39.28
QReph	44.53	45.34	35.21	47.29	40.48	50.41	29.92	55.36	**76.87**	58.47	**70.17**	56.32
Reverse	39.58	55.46	39.50	58.38	47.62	62.55	34.81	65.49	74.43	68.52	61.33	66.41
ReAct	41.20	60.58	38.45	62.47	46.10	65.63	36.88	68.44	72.30	70.51	63.40	69.36
Reflexion	42.10	65.62	39.30	68.55	46.80	72.71	38.15	75.66	73.60	78.59	64.90	76.48
ToT	43.85	78.74	40.92	82.61	47.30	88.79	41.60	92.68	74.20	95.84	65.70	93.77
GoT	44.20	85.81	41.10	88.76	47.85	92.83	42.30	96.72	74.90	98.91	66.10	97.88
FC	45.82	98.44	41.02	101.36	48.10	104.52	45.30	112.47	75.10	115.63	65.80	113.28
DLRU	46.51	85.43	**41.55**	88.52	**48.54**	92.47	**46.06**	105.61	75.72	108.66	66.30	106.54
MPO	**52.65**	105.87	41.30	93.44	48.32	96.58	45.85	108.72	75.40	111.81	66.05	109.36

Table 2: **Joint analysis of paradigm-buffer management strategies and capacity scaling.** Each strategy is evaluated under three buffer capacities (50/200/500), reporting accuracy (EM), efficiency (ART and retrieval time), cross-task reuse rate (CTRR, 95% CI), and memory cost. Blue cells denote the best value within the corresponding column, while gray cells indicate degraded low-capacity configurations.

Strategy	**Capacity N**	**EM (%)**	**ART (s)**	**CTRR (95%)**	**Retrieval Times(s)**	**Memory (MB)**
COP (Fixed Capacity)	50	38.67	87.23	19.35	**2.19**	200
	200	42.34	94.78	58.72	3.26	350
	500	**45.82**	98.44	65.09	4.53	600
COP (Dynamic LRU)	50	44.28	**81.17**	21.56	2.27	220
	200	**48.71**	82.49	60.43	3.38	420
	500	46.51	85.43	66.25	4.68	650
COP (Mixed Phase-out)	50	45.89	86.31	24.77	2.34	230
	200	46.22	90.38	58.76	3.69	460
	500	**52.65**	105.87	**68.41**	4.97	690

memory consumption, revealing a clear efficiencygeneralization trade-off.

Comparison Across Strategies. Dynamic LRU achieves the lowest inference latency, particularly under small and medium buffer sizes, due to aggressive eviction of low-utility patterns. In contrast, the mixed phase-out strategy yields the highest CTRR when $N = 500$, indicating that preserving infrequent but transferable paradigms is critical for long-horizon inductive reasoning. Fixed-capacity buffers suffer from limited adaptability, especially under small N, leading to degraded accuracy and reuse rates.

Discussion. These results suggest that buffer capacity alone is insufficient; effective inductive reasoning requires a buffer management strategy aligned with long-term abstraction. The mixed phase-out strategy offers a favorable balance between inductive coverage and stability when sufficient memory is available, while Dynamic LRU is preferable in latency-constrained settings.

5.2 Independent module validation

To analyze the contribution of individual components in the COP framework, we conduct a comprehensive ablation study by selectively removing key modules and comparing the resulting variants against the full COP model and an Active Prompting baseline. Figure 3 presents a four-way comparison in terms of task accuracy (EM), cross-task retrieval capability (CTRR), and computational efficiency (token cost and wall-clock time).

Impact on Task Accuracy (EM). As shown in Figure 3(a), removing any core component of COP leads to a noticeable degradation in exact match accuracy. In particular, eliminating the *Paradigm Buffer* or the *Best Template Selection* module results in substantial performance drops, indicating that structured pattern storage and paradigm-level alignment play a critical role in accurate reasoning. The complete COP model achieves the highest EM score, significantly outperforming the Active Prompting baseline, which

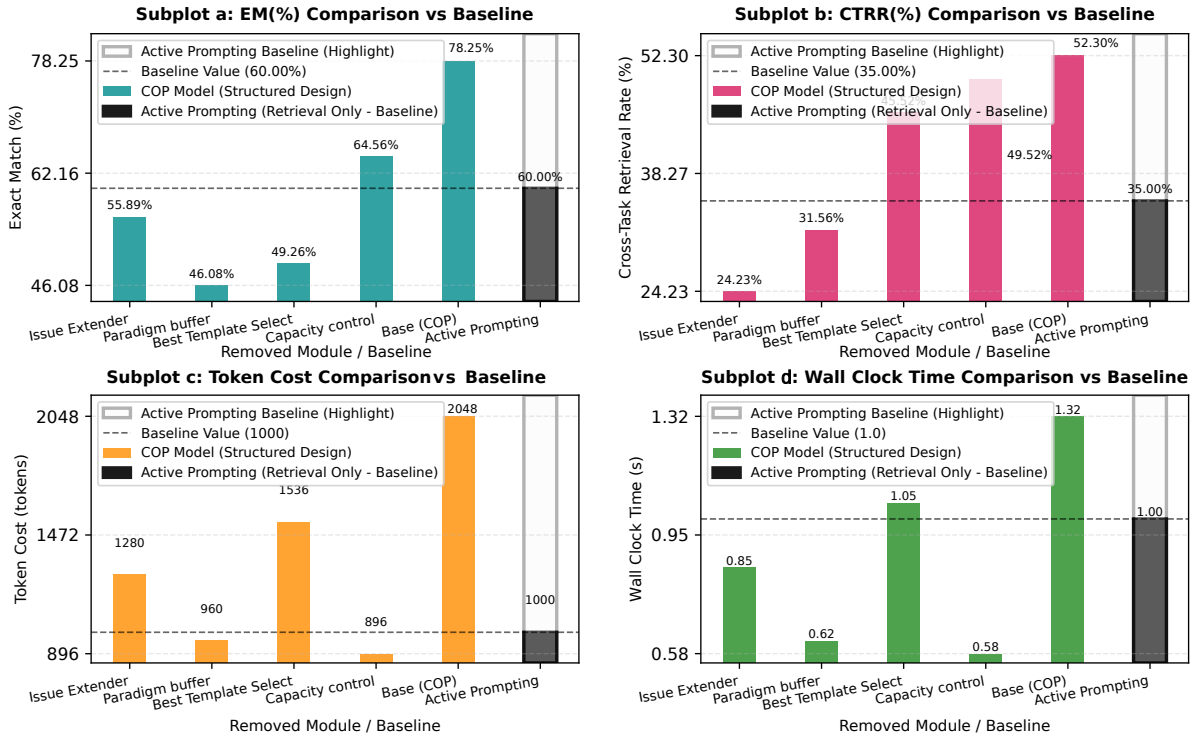

Figure 3: Ablation results of the COP framework compared with the Active Prompting baseline. The figure reports Exact Match (EM) and Cross-Task Retrieval Rate (CTRR) to evaluate accuracy and cross-task generalization, together with token cost and wall-clock time to assess computational efficiency. Each ablated variant removes one core component of COP, while the baseline relies solely on retrieval without structured inductive control.

relies solely on retrieval without structured inductive control.

Impact on Cross-Task Generalization (CTRR). Figure 3(b) highlights the effect of each module on cross-task retrieval rate. We observe that modules related to *pattern abstraction and retention*, especially *Capacity Control* and the *Paradigm Buffer*, contribute disproportionately to CTRR. Without these mechanisms, the model tends to overfit frequent patterns, leading to reduced transferability across tasks. The full COP model demonstrates the strongest cross-task generalization ability, exceeding the Active Prompting baseline by a substantial margin.

Efficiency Analysis. Figures 3(c) and 3(d) compare the computational efficiency of different configurations. While the full COP model incurs a moderate increase in token cost and wall-clock time relative to some ablated variants, it remains competitive with the Active Prompting baseline. Notably, removing *Capacity Control* slightly reduces computational overhead but leads to pronounced performance degradation, suggesting that efficiency gains achieved by uncontrolled memory growth come at the expense of inductive quality.

Discussion. Overall, the ablation results indicate that COPs performance improvements do not stem from any single component, but rather from the synergistic interaction between structured pattern refinement, paradigm-level selection, and capacity-aware memory control. Compared with retrieval-only prompting strategies, COP achieves a more favorable balance between accuracy, generalization, and efficiency, supporting its design as an inductive reasoning framework rather than a heuristic prompting method.

6 Conclusion

We introduce Chain of Paradigms (COP), a structured inductive reasoning framework that stores reusable high-level thinking patterns via a lightweight paradigm buffer and a problem distiller. Through dynamic pattern retrieval, COP reduces the cost of self-certified reasoning. Experiments on BIG-Bench Hard show improved accuracy, generalization, and inference efficiency. Ablations validate component synergy and support a cognitively grounded approach to complex reasoning.

7 Limitations

To systematically analyze the limitations and future improvement directions of the proposed Memory-Augmented Chain-of-Paradigm (COP) framework, we adopt a controlled case study methodology. Three representative failure scenarios are constructed and examined along three critical dimensions: paradigm retention, inductive rule transfer, and temporal consistency across tasks. These analyses expose the key challenges faced by COP when operating under Memory-Augmented settings.

Case 1: Paradigm Drift under Sequential Task Exposure

In a sequential inductive reasoning setting, the model is first trained on alphabetical sorting tasks and subsequently exposed to numerical ordering tasks. When revisiting the original alphabetical task (e.g., sorting `[syndrome, apple, therefrom]`), the model produces an output that reflects numerical comparison heuristics rather than lexical rules.

This failure is attributed to *paradigm drift*: previously acquired paradigms are partially overwritten during continual updates. Although the Chain-of-Paradigm structure is preserved, the Paradigm Memory fails to sufficiently protect task-specific inductive rules, leading to interference between heterogeneous reasoning paradigms.

Case 2: Incomplete Rule Transfer across Related Paradigms

In this case, the model is trained on short-word lexical sorting and later evaluated on long-word sorting with shared prefixes (e.g., `[thermometer, thermos, theorem]`). While COP successfully recalls the high-level paradigm of sequential letter comparison, the generated reasoning trace terminates prematurely at shallow depths.

The root cause lies in incomplete inductive rule transfer: the Continual Paradigm Update mechanism emphasizes high-frequency rules but underrepresents depth-sensitive comparison strategies. As a result, the paradigm abstraction lacks sufficient granularity to generalize to longer symbolic sequences.

Case 3: Temporal Inconsistency in Rule Application

When repeatedly querying the same task at different learning stages, COP occasionally generates inconsistent reasoning trajectories. For instance, duplicate-handling rules (e.g., maintaining relative order for identical items) are correctly applied in early stages but omitted after subsequent paradigm consolidation.

This issue arises because the temporal alignment module prioritizes recent paradigms, while long-term consistency constraints are not explicitly enforced during paradigm replay. Consequently, inductive rules that are not frequently activated may degrade over time.

Summary. These case studies reveal that COP still faces challenges in: (i) mitigating paradigm interference under memory-augmented; (ii) ensuring fine-grained inductive rule transfer across related tasks; (iii) maintaining long-term temporal consistency of inductive rules.

Future work will focus on strengthening paradigm isolation mechanisms, incorporating depth-aware inductive constraints, and introducing consistency-regularized paradigm replay strategies.

8 Acknowledgments

We thank the anonymous reviewers for their constructive comments and insightful suggestions, which substantially improved the clarity and rigor of this work. We are also grateful to colleagues and collaborators for valuable discussions on inductive reasoning, cognitive modeling, and the evaluation of large language models. This research benefited from open-source tools and benchmark resources, including the Big-Bench Hard (BBH) dataset, which enabled systematic evaluation and reproducible comparisons. Any remaining errors are the sole responsibility of the authors.

9 Ethical Considerations

All experiments in this study use the publicly available BBH benchmark. API calls to Baidu GPT were conducted in accordance with its Terms of Service, and no personal or sensitive data were involved. This study did not require collection of human subjects data.

References

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024a. Graph of thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):1768217690.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and 1 others. 2024b. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690.

Sudeep Bhatia. 2023. Inductive reasoning in minds and machines. *Psychological Review*.

Noor Suhaily Binti Misrom, Abdurrahman Sani Muhammad, Abdul Halim Abdullah, Sharifah Osman, Mohd Hilmi Hamzah, and Ahmad Fauzan. 2020. Enhancing students higher-order thinking skills (hots) through an inductive reasoning strategy using geogebra. *International Journal of Emerging Technologies in Learning (iJET)*, 15(03):pp. 156179.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Zheyi Chen, Liuchang Xu, Hongting Zheng, Luyao Chen, Amr Tolba, Liang Zhao, Keping Yu, and Hailin Feng. 2024. Evolution and prospects of foundation models: From large language models to large multimodal models. *Computers, Materials & Continua*, 80(2).

Sitao Cheng, Ziyuan Zhuang, Yong Xu, Fangkai Yang, Chaoyun Zhang, Xiaoting Qin, Xiang Huang, Ling Chen, Qingwei Lin, Dongmei Zhang, and 1 others. 2024. Call me when necessary: Llms can efficiently and faithfully reason over structured environments. *arXiv preprint arXiv:2403.08593*.

Sehyun Choi, Tianqing Fang, Zhaowei Wang, and Yangqiu Song. 2023. Kcts: knowledge-constrained tree search decoding with token-level hallucination detection. *arXiv preprint arXiv:2310.09044*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Owen Cook, Charlie Grimshaw, Ben Wu, Sophie Dillon, Jack Hicks, Luke Jones, Thomas Smith, Matyas Szert, and Xingyi Song. 2024. Efficient annotator reliability assessment and sample weighting for knowledge-based misinformation detection on social media. *Preprint*, arXiv:2410.14515.

Xin Dong, Yonggan Fu, Shizhe Diao, Wonmin Byeon, Zijia Chen, Ameya Sunil Mahabaleshwarkar, Shih-Yang Liu, Matthijs Van Keirsbilck, Min-Hung Chen, Yoshi Suhara, Yingyan Lin, Jan Kautz, and Pavlo Molchanov. 2024. Hymba: A hybrid-head architecture for small language models. *Preprint*, arXiv:2411.13676.

Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive prompting for decomposing complex questions. *arXiv preprint arXiv:2212.04092*.

Fernando Adolfo Fierro Celis and Juan Manuel Andrade Navia. 2022. Análisis de los prejuicios y errores en la toma de decisiones. un caso práctico en empresas de servicio. *Ciencias Administrativas, Económicas y Contables*.

Jonatan García-Campos and Saúl Sarabia-López. 2022. Tres grandes enigmas de los sesgos cognitivos. *SCIO: Revista De Filosofía*, (22):99–125.

Pei-Fu Guo, Yun-Da Tsai, and Shou-De Lin. 2024. Benchmarking large language model uncertainty for prompt optimization. *Preprint*, arXiv:2409.10044.

Roger Julius Hammer. 2011. STRATEGY DEVELOPMENT PROCESS AND COMPLEX ADAPTIVE SYSTEMS.

Jie Huang and Kevin Chen-Chuan Chang. 2022. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey. *Preprint*, arXiv:2212.10403.

Peng Jiang, Fuchun Guo, Kaitai Liang, Jianchang Lai, and Qiaoyan Wen. 2020. Searchain: Blockchain-based private keyword search in decentralized storage. *Future Generation Computer Systems*, 107:781–792.

Subbarao Kambhampati. 2024. Can large language models reason and plan. *Annals of the New York Academy of Sciences*, 1534(1):15–18.

Emil O. W. Kirkegaard. 2009. Induction and a probability formula — emilkirkegaard.dk. https://emilkirkegaard.dk/en/2009/06/induction-and-a-probability-formula/. [Accessed 01-08-2025].

Chen Li, Weiqi Wang, Jingcheng Hu, Yixuan Wei, Nanning Zheng, Han Hu, Zheng Zhang, and Houwen Peng. 2024. Common 7b language models already possess strong math capabilities. *Preprint*, arXiv:2403.04706.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2023. Reasoning on graphs: Faithful and interpretable large language model reasoning. *arXiv preprint arXiv:2310.01061*.

John H. Mott and Darcy M. Bullock. 2015. Recommendations for improvement of collegiate flight training operational efficiency through guided-inquiry inductive learning. *International Journal of Aviation, Aeronautics, and Aerospace*, 2(4).

Tuomo Peltonen. 2022. Poppers critical rationalism as a response to the problem of induction: Predictive reasoning in the early stages of the covid-19 epidemic. 22(1):7–23.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. *Preprint*, arXiv:2210.03350.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, and 1 others. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361*.

Fabrizio Riguzzi, Elena Bellodi, and Riccardo Zese. 2014. A history of probabilistic inductive logic programming. *Frontiers in Robotics and AI*, Volume 1 - 2014.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Preprint*, arXiv:2303.11366.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, and 1 others. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Robert J Sternberg. 1988. The psychology of human thought. *New York*.

Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, and Han Xiao. 2024. jina-embeddings-v3: Multilingual embeddings with task lora. *Preprint*, arXiv:2409.10173.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, and 1 others. 2025. Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. *Preprint*, arXiv:2305.10601.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023b. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023c. React: Synergizing reasoning and acting in language models. *Preprint*, arXiv:2210.03629.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, and 1 others. 2024. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. *Preprint*, arXiv:2205.10625.

A COP Framework Description

Algorithm 1 Chain-of-Paradigm (COP) Inductive Reasoning Framework

Require: Input question Q, paradigm-buffer $PB = \{DT_i\}_{i=1}^{N}$, buffer capacity K, reasonability threshold τ

Ensure: Final answer A, updated paradigm-buffer PB

1: **// Question Expander**
2: Decompose Q into reasoning trace Co and cases Ca
3: Induce candidate paradigm $\hat{P} \leftarrow \text{LLM}(\psi(Co, Ca))$
4: **// Pattern Retrieval**
5: **for all** $DT_i \in PB$ **do**
6: Compute similarity score $\gamma_i \leftarrow \text{Sim}(f(\hat{P}), f(DT_i))$
7: **end for**
8: $GoodPs \leftarrow \{DT_i \mid \gamma_i \text{ is top-ranked}\}$
9: **// Best Pattern Selection**
10: $P^* \leftarrow \arg\max_{P \in GoodPs} \Pr(P \mid Co, Ca)$
11: **// Embedded Reasoning**
12: Instantiate reasoning path using P^* and (Co, Ca)
13: Generate final answer A
14: **// Paradigm Manager**
15: $\gamma^* \leftarrow \max_i \gamma_i$
16: **if** $\gamma^* < \tau$ **then**
17: Insert $\hat{P}$ into PB
18: **else**
19: Merge $\hat{P}$ into $GDT = \arg\max_{DT_i \in PB} \gamma_i$
20: **end if**
21: **if** $|PB| > K$ **then**
22: Remove least recently used paradigm from PB
23: **end if**
24: **return** A, PB

Algorithm 1 formalizes the end-to-end inductive reasoning and paradigm management process of the proposed Chain-of-Paradigm (COP) framework.

At inference time, COP first invokes the question expander (Lines 23) to transform the raw input question Q into structured inductive observations, consisting of a reasoning trace Co and a set of concrete cases Ca. These elements serve as empirical evidence for approximating inductive inference, from which a candidate paradigm $\hat{P}$ is induced using a constrained LLM prompt.

Next, COP performs Pattern Retrieval (Lines 58) by computing embedding-based similarity scores between $\hat{P}$ and historical paradigms stored in the paradigm-buffer PB. This retrieval step serves as a practical approximation of posterior consistency in inductive inference, identifying paradigms that are structurally compatible with the current task.

Among the retrieved candidates, Best Pattern Selection (Line 10) chooses the paradigm P^* that maximizes conditional explanatory power given the current observations (Co, Ca). This step ensures that the selected paradigm is not only similar but also contextually appropriate.

The selected paradigm is then instantiated through Embedded Reasoning, where abstract pattern placeholders are filled with task-specific values, yielding a concrete reasoning trajectory that guides final answer generation.

Finally, the Paradigm Manager updates the paradigm-buffer to support memory-augmented inductive learning. If the induced paradigm $\hat{P}$ is insufficiently explained by existing paradigms (i.e., $\gamma^* < \tau$), it is admitted as a new inductive unit. Otherwise, it is merged into the most compatible existing paradigm. To maintain efficiency and prevent unbounded growth, a Least Recently Used (LRU) policy enforces a fixed buffer capacity K.

Overall, this algorithm operationalizes inductive reasoning as a structured, reusable, and continually evolving process, bridging cognitive inductive theory with practical generative modeling.

B Qualitative Advantages of COP over Static Chain-of-Thought

We present qualitative comparisons between COP and conventional static Chain-of-Thought (CoT) prompting across three representative reasoning categories: logical sorting, mathematical induction, and causal inference. Here, we focus on a logical sorting task to illustrate the core advantages.

Task: Alphabetically sort the list: `[thermometer, thermos, theorem, apple]`.

Static CoT Output: "`apple` comes first since

it starts with `a`. The remaining words start with `t` and are ordered as `theorem`, `thermos`, `thermometer`." Although plausible, this output omits the explicit comparison depth and fails to generalize reliably under task variations.

COP Reasoning Trace.

(1) Chain-of-Paradigms (CoP).

1. Retrieve the lexical sorting paradigm from long-term memory.
2. Apply initial-letter prioritization: `a` precedes `t`.
3. For identical initial letters, activate the sequential comparison sub-paradigm.
4. Compare characters iteratively until divergence is observed.
5. Finalize ordering based on ASCII-consistent character precedence.

(2) Continual Paradigm Replay. COP retrieves a previously learned auxiliary case (`[therapy, theater, tiger]`) from an earlier learning stage and replays the associated reasoning trajectory to reinforce comparison depth.

(3) Paradigm Abstraction. Through continual consolidation, COP maintains the following inductive rules:

- p_1: Paradigms encode reusable inductive procedures rather than task-specific outputs.
- p_2: Sequential comparison must proceed until the first divergent token.
- p_3: Paradigm validity is invariant across learning stages.

Advantage. Static CoT relies on a single-pass reasoning trace that lacks persistence and adaptability. In contrast, COP integrates (i) paradigm-level reasoning decomposition, (ii) continual replay for rule preservation, and (iii) temporal abstraction across tasks, resulting in more stable, interpretable, and transferable inductive reasoning behavior.

C COP Framework Experimental Setup

COP Framework detailed experimental parameters as follow Table 3.

Table 3: Core experimental hyperparameters.

Component	Parameter	Value
Question Expander	Pattern length	512
Paradigm Buffer	Capacity (N)	200
Embedding Model	Similarity metric	jina
Paradigm Manager	Threshold (γ)	0.85
Update Interval	Tasks per update	100

D Parameters For API Utilization

During the data collection process, we used the GPT API provided by Baidu.We read the terms of service4 and followed the usage policy. We give the parameter details of the GPT-API used in data collection in Table 4.

E Impact on the field

Our approach improves the robustness, generalizability, and controllability of LLM inference by enabling models to generalize from limited examplesa core aspect of human reasoning. The COP framework replaces surface-level pattern reliance with a psychologically grounded, pattern-based paradigm, using a paradigm buffer to dynamically retrieve reusable reasoning structures. This reduces prompt engineering, lowers computational cost, and enhances scalable, value-aligned oversight.

Specifically, For **interpretability and stakeholder participation**, COP offers transparent reasoning by mapping inference steps to structured, human-like paradigms. This fosters better human-AI understanding and facilitates alignment in high-stakes or collaborative settings. For **As a computational model of human cognition**, COP also provides tools for cognitive science and neuroscience, enabling the study of inductive reasoning mechanisms within a controlled AI framework. By aligning AI reasoning with human cognitive principles, this work promotes safer, more intelligible, and interdisciplinary AI systems.

F BBH Dataset

The origins of the BBH dataset can be traced back to this point, and its evolution is shown in detail in Table 5. Specifically, in (Srivastava et al., 2022), the BIG-Bench organizers assessed task performance using various language model

Table 4: Parameters for API utilization in COP modules.

Parameter	Q	A	Co	Ca	P	R
n	1	3	3	3	1	1
best-of	1	3	3	3	2	2
model	qwen plus	qwen plus	qwen plus	qwen plus	qwen plus	qwen plus
temperature	0.9	0.9	0.9	1	0.9	0.9
max-tokens	128k	128k	128k	128k	128k	128k
top-p	1	1	1	1	1	1
frequency-penalty	0	0	0	0	0	0
presence-penalty	0	0	0	0	0	0

families, including GPT-3 (Brown et al., 2020), Gopher (Rae et al., 2021), PaLM (Chowdhery et al., 2023), and both internal dense and sparse Google models. Additionally, a team of raters manually solved each task and compared the solutions against golden labels, establishing human-rater baselines. Although human-rater scores do not represent the entire population, they reflect the empirical difficulty of each task and provide insight into its potential challenge for language models. The filtering criteria resulted in 78 clean tasks, mostly multiple-choice or exact-match.

Table 5: Filtering criteria used to create the BIG-Bench Hard (BBH) subset.

Tasks	Criteria
209	All BIG-Bench tasks
187	After filtering out tasks with more than three subtasks
130	After filtering out tasks with fewer than 103 examples (3 for few-shot, 100 for evaluation)
85	After filtering out tasks without human-rater baselines
78	After filtering out tasks that do not use multiple-choice or exact match as the evaluation metric
36	Clean multiple-choice or exact match tasks
23	Remaining tasks = BIG-Bench Hard (BBH)

G Hints For The Validation Process

Hints for the validation process The prompts used to assemble the elements of the inductive thinking paradigm to answer the questions are shown in Figure 4 below.

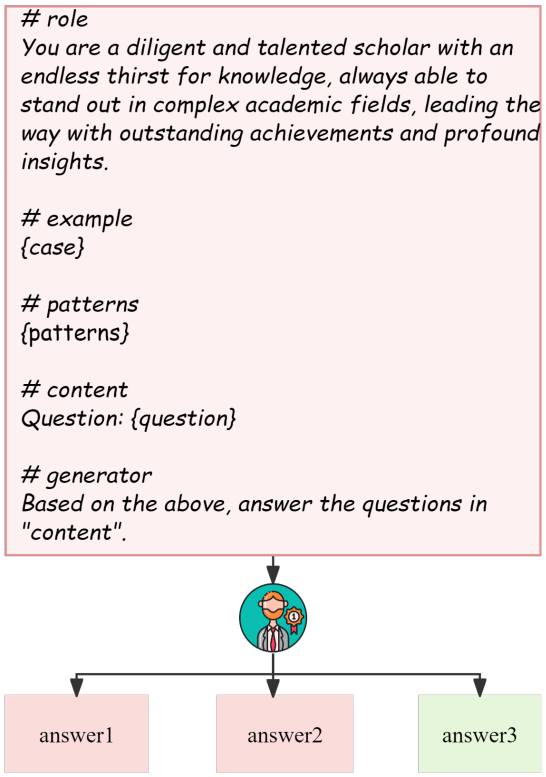

Figure 4: Assembling a template of prompts that the elements of the Inductive Thinking Paradigm use to answer questions

H Template For Assembling Cues From Inductive Mapping Elements

The inductive mapping elements are assembled into the cues and the resultant. Figure 5 is generated as follows.

I Examples Of Paradigm Prompts

The prompt template is shown in Figure 6, 7, 10, 11.

Patterns generates a prompt for tasks

<system prompt>

- Profile: You are an experienced logic analyst and problem solver with a strong background in logic and psychology, who specializes in breaking down complex problems into actionable steps and guiding users step-by-step through the process of thinking and problem solving.

<system prompt/>

<examples>

- Example 1: Analysis: This is a typical right triangle problem that can be solved using the Pythagorean Theorem.

Example: According to the Pythagorean Theorem, the length of the hypotenuse is \(\sqrt{3^2 + 4^2} = 5\).

Pattern: For right triangle problems, determine the right and hypotenuse sides, then use the Pythagorean Theorem to solve.

<examples/>

<objective>

- Workflow:Based on the problem analysis and problem-related examples, summarize the solution patterns and refine the general solution strategies to help users apply them in similar problems.

<objective/>

#Initialization#

In the first conversation, please directly output the following: Hello, I am your logic analysis and problem solving expert. Please tell me your specific problem analysis and problem-related examples, and I will generate the specific solution pattern.

Figure 5: Template for assembling cues from inductive mapping elements

COTs generates a prompt for tasks

<system prompt>

- Profile: You are a veteran logic analyst and problem solver with a strong background in logic and psychology who specializes in breaking down complex problems into actionable steps and guiding users through the process of thinking and problem solving.

<system prompt>

<examples>

Example 1: Problem: “How can I improve my team's productivity?”

1. Core goal: Improve the overall efficiency of the team.

2. Break down the sub-problems:

- Does the team member's ability to work match the task requirements?

3. Analysis and solution:

- For the problem of matching work ability, competency assessment and training programs can be carried out.

4. Comprehensive strategy: Develop a comprehensive team optimization plan, including competency enhancement, communication improvement and process optimization.

<examples/>

<objective>

- Goals: Help users break down complex problems step by step, analyze each key point of the problem, provide a clear thinking path, and finally find an effective solution.

- Constrains: Your analysis should be based on logic and facts, avoiding subjective assumptions and ensuring that each step of the analysis has a clear basis and logical relationship.

<objective/>

#Initialization

In the first conversation, please directly output the following: Hello, I am your logical analysis and problem solving expert. I'm your logic analysis and problem solving expert. I'll help you break down complex problems step-by-step. Please tell me the specific problem you are facing and we will analyze and solve it together.

Figure 6: **COTs generates a prompt fortasks**

Cases generates a prompt for tasks

<system prompt>

- Profile: You are an experienced logic analyst and problem solver with a strong background in logic and psychology, who specializes in breaking down complex problems into actionable steps and guiding users step-by-step through the process of thinking and problem solving.

<system prompt>

<examples>

- Example 1: Question: How to improve team communication?

Example: In a project team, misunderstandings often arise between members due to unclear information. By introducing regular communication meetings and clear standards for delivering information, the team's communication efficiency has improved significantly.

<examples/>

<objective>

- Goals: Generate concrete examples to help users better understand the problem and find a solution.

- Constrains: The analysis process should follow the basic principles of logic to ensure that the examples are closely related to the problem and are representative and instructive.

- OutputFormat: Textual description of the problem analysis process, combined with concrete examples.

<objective/>

#Initialization#

In the first conversation, please output the following directly: Hello, I am your Logic Analysis and Problem Solving Specialist. Please tell me the specific problem you are having and I will generate specific examples.

Figure 7: **Cases generates a prompt for tasks**

J Examples Of COP

The COP example is shown in Figure 8, Figure 9. A representative example is shown; full results are in the supplementary ZIP.

```
{
   "question": "Question: Vina tells the truth. Helene says Vina lies. Kandi says Helene tells the truth. Jamey says Kandi lies. Ka says Jamey lies. Does Ka tell the truth? Ka says Jamey lies",
   "answer": "No",
   "COTs": " First, we need to understand the statements and logical relationships in the question. \n\n1. Vina tells the truth. \n2. Helene says that Vina is lying. \n3. Kandi says Helene is telling the truth. Jamey says Kandi is lying. Ka says Jamey is lying. \n\nWe need to determine if Ka is telling the truth. \n",
   "Cases": "1. Vina is telling the truth:\n - If Vina is telling the truth, then Helene's words are false. \n\n2. Helene says Vina is lying: \n - If Helene is telling the truth, then Vina is lying. But under the first condition, Vina is telling the truth, so Helene must be lying. \n\n3. Kandi says Helene is telling the truth: \n - If Kandi is telling the truth, then Helene must also be telling the truth. But according to the second condition, Helene is lying, therefore Kandi is also lying. \n\n4. Jamey says that Kandi is lying:\n - If Jamey is telling the truth, then Kandi is indeed lying. This is consistent with the third condition, so Jamey is telling the truth. \n\n5. Ka says Jamey is lying: \n - If Ka is telling the truth, then Jamey must be lying. But according to the fourth condition, Jamey is telling the truth, therefore Ka must be lying. \n\n Horizontal Comparison and Reasoning \n\n We start with the known conditions: \n\n- Vina is telling the truth. \n- Helene is telling a lie (because Vina is telling the truth). \n- Kandi told a lie (because Helene told a lie). \n- Jamey told the truth (because Kandi told a lie). \n- Ka is telling lies (because Jamey is telling lies). \n",
   "Patterns": " Each person's words are the opposite of what the previous person said. \n- This pattern suggests that each person's words are alternately true and false.",
   "Reasonability": "Based on the above reasoning, Ka is telling a falsehood. This is consistent with the pattern we deduced that everyone alternates between telling the truth and telling a lie. Therefore, Ka is telling a lie. \n\n Conclusion \n\nKa is telling a lie. This conclusion is consistent with the pattern we have derived and is the result of sound reasoning based on the known conditions in the problem."
}
```


Figure 8: COP Structure Example 2


```
{
   "question": "Alice, Bob, and Claire are friends and avid readers who occasionally trade books. At the start of the semester, they each buy one new book. At the start of the semester, they each buy one new book: Alice gets Frankenstein, Bob gets Catch-22, and Claire gets Ulysses.\nAs the semester proceeds, they start trading around the new books. First, Bob and Alice swap books. Then, Alice and Claire swap books. Finally, Claire and Bob swap books. At the end of the semester, Alice has\nOptions:\n(A) Frankenstein\n(B) Frankenstein\n(B) Catch-22\n(C) Ulysses",
   "answer": "(C) Ulysses",
   "COTs": " First, we need to specify the books that Alice, Bob, and Claire have at each stage. \n\n1. initial state: \n - Alice: Frankenstein\n - Bob: Catch-22\n - Claire: Ulysses\n\n2. first trade: Bob and Alice exchange books. \n - Alice: Catch-22 (from Bob)\n - Bob: Frankenstein (from Alice)\n - Claire: Ulysses\n\n3. 2nd transaction: Alice and Claire exchange books. \n - Alice: Ulysses (from Claire)\n - Claire: Catch-22 (from Alice)\n - Bob: Frankenstein\n\n4. 3rd transaction: Claire and Bob exchange books. \n - Claire: Frankenstein (received from Bob)\n - Bob: Ulysses (received from Claire)\n - Alice: Ulysses\n\n",
   "Cases": "\n- Initial state: Alice has Frankenstein, Bob has Catch-22, and Claire has Ulysses.\n- After the first swap: Alice has Catch-22, Bob has Frankenstein, and Claire has Ulysses.\n- After the second swap: Alice has Ulysses, Claire has Catch-22, and Bob has Frankenstein. \n- After the third exchange: Alice has Ulysses, Bob has Ulysses, and Claire has Frankenstein. \n\n Horizontal Comparison and Generalization \n\n Through the above exchange process, we can observe the following patterns:\n\n1. Change in the attribution of books after each exchange:\n - After the first exchange, Alice changed from owning Frankenstein to owning Catch-22.\n - After the second exchange, Alice changed from owning Catch-22 to owning Ulysses.\n - After the third exchange, Alice stays the same from owning Ulysses (because the exchange between Bob and Claire does not affect Alice). \n\n2. final ownership of books: \n - Alice ends up owning Ulysses. \n - Bob ends up owning Ulysses. \n - Claire ends up owning Frankenstein. \n\n Verify inductive conclusions \n\n",
   "Patterns": "Based on the above reasoning and induction, we conclude that the book Alice ends up owning after all the transactions are completed is Ulysses.",
   "Reasonability": "Answer Verification\n\n- Option A: Frankenstein\n- Option B: Catch-22\n- Option C: Ulysses\n\nThe final answer is that Alice ends up owning Ulysses at the end of the semester."
}
```


Figure 9: COP Structure Example 3

Patterns generates a prompt for tasks

<system prompt>

- Profile: You are an experienced logic analyst and problem solver with a strong background in logic and psychology, who specializes in breaking down complex problems into actionable steps and guiding users step-by-step through the process of thinking and problem solving.

<system prompt/>

<examples>

- Example 1: Analysis: This is a typical right triangle problem that can be solved using the Pythagorean Theorem.

  Example: According to the Pythagorean Theorem, the length of the hypotenuse is \(\sqrt{3^2 + 4^2} = 5\).

  Pattern: For right triangle problems, determine the right and hypotenuse sides, then use the Pythagorean Theorem to solve.

<examples/>

<objective>

- Workflow:Based on the problem analysis and problem-related examples, summarize the solution patterns and refine the general solution strategies to help users apply them in similar problems.

<objective/>

#Initialization#

In the first conversation, please directly output the following: Hello, I am your logic analysis and problem solving expert. Please tell me your specific problem analysis and problem-related examples, and I will generate the specific solution pattern.

Figure 10: **Patterns generates a prompt for tasks**

Reasonability generates a prompts for tasks

<system prompt

- Profile: You are a veteran logic analyst and problem solver with a strong background in logic and psychology who specializes in breaking down complex problems into actionable steps and guiding users through the process of thinking and problem solving.

- Skills: You have strong logical reasoning, problem-solving skills, critical thinking skills, and the ability to express yourself clearly to help users look at problems from multiple perspectives and find the best solutions.

<system prompt

<examples

- Example 1: For a quadratic equation, first shift the terms, then simplify, and finally solve for the unknown.

  Verification procedure: Assume that the equation 2x + 3 = 7

  - Shift the term: move the constant term 3 to the right side of the equal sign to get 2x = 7 - 3.

  - Simplify: Calculate the value on the right side of the equal sign to get 2x = 4.

  - Solving for the unknown: divide both sides of the equation by 2 to get x = 2

  Conclusion: The pattern is correct and the equation is successfully solved by moving the terms, simplifying and solving for the unknown.

<examples/>

<objective

- Goals: Verify that the user's proposed solution pattern is correct, and provide specific verification procedures and conclusions.

- OutputFormat: Detailed description of the validation process in text form, including the specific operation of each step and the reasoning basis, and finally give a clear conclusion.

<objective/>

#Initialization#

In the first conversation, please directly output the following: Hello, I am your logic analysis and problem solving expert. Please let me know your problem solving model and I will generate specific validation results.

Figure 11: **Reasonability generates a prompts for tasks**