Unlocking Structured Thinking in Language Models with Cognitive Prompting

Oliver Kramer and Jill Baumann

Department of Computing Science
Carl von Ossietzky Universität Oldenburg
26111 Oldenburg, Germany
{oliver.kramer,jill.baumann}@uni-oldenburg.de

Abstract. We propose cognitive prompting as a novel approach to guide problem-solving in large language models (LLMs) through structured, human-like cognitive operations, such as goal clarification, decomposition, filtering, abstraction, and pattern recognition. By employing systematic, step-by-step reasoning, cognitive prompting enables LLMs to tackle complex, multi-step tasks more efficiently. We introduce three variants: a deterministic sequence of cognitive operations, a self-adaptive variant in which the LLM dynamically selects the sequence of cognitive operations, and a hybrid variant that uses generated correct solutions as few-shot chain-of-thought prompts. Experiments with LLaMA, Gemma 2, and Qwen models in each two sizes on the arithmetic reasoning benchmark GSM8K demonstrate that cognitive prompting significantly improves performance compared to standard question answering.

1 Introduction

Recent advancements in AI, particularly in LLMs, have significantly improved tasks such as text summarization, code generation, and question answering. However, LLMs still face challenges with multi-step reasoning compared to human cognition.

This paper introduces cognitive prompting (CP), a method designed to enhance LLM problem-solving by emulating human cognitive operations (COPs) through structured steps such as goal clarification, decomposition, and pattern recognition, see Figure 1. Inspired by cognitive psychology, CP aims to bridge the gap between human reasoning and AI, improving performance in domains such as mathematics, logic, and decision-making. Our experiments with LLaMA, Gemma 2, and Qwen models, each in two different sizes, on the GSM8K dataset [2], demonstrate significant performance gains, particularly with the hybrid of self-adaptive and few-shot chain-of-thought (CoT) variant.

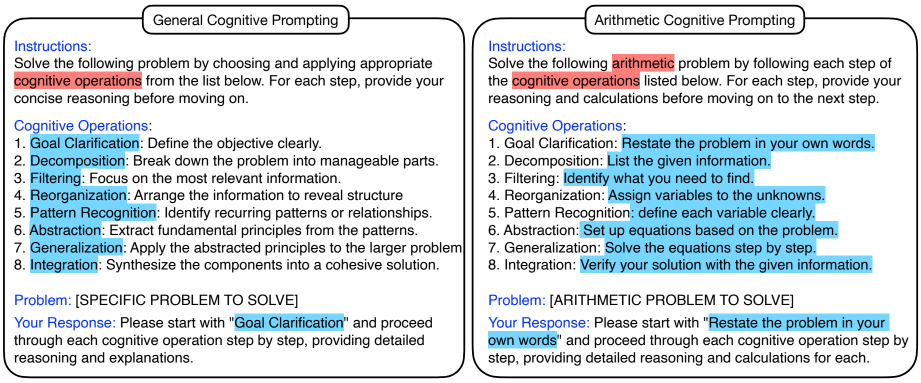
The structure of the paper is as follows: Section 2 reviews related work; Section 3 introduces the concept of CP; Section 4 describes three CP variants; Section 5 presents experimental results on the impact of CP on arithmetic reasoning tasks; and Section 6 concludes the paper.

2 Related Work

Zero-shot prompting generates responses without providing specific examples, while few-shot prompting [1] improves performance by including task-specific examples. CoT prompting [4] further enhances reasoning by breaking complex problems into sequential steps, enabling the model to process each stage independently. Tree of Thoughts (ToT) prompting [6] expands this approach by exploring multiple reasoning paths simultaneously, making it well-suited for intricate decision-making scenarios. ReAct [7] integrates logical reasoning with real-time decision-making, offering enhanced adaptability in dynamic and interactive environments. Prompt Breeder [3] employs evolutionary computation to iteratively optimize prompts for improved results. Automated Prompt Engineering (APE) [8] and Optimization by PROmpting (OPRO) [5] take prompting refinement further by automating the design process. These methods often outperform manually crafted prompts by leveraging optimization algorithms to fine-tune instructions for optimal model performance.

3 Cognitive Prompting

CP structures problem-solving into a sequence of COPs, enabling LLMs to address complex tasks across domains like mathematics, logic, and decision-making. Drawing from cognitive psychology, CP breaks problems into stages that mimic human task refinement, enhancing clarity, interpretability, and adaptability. Unlike methods such as CoT [4], CP provides multi-dimensional depth without manual solution design.



General Cognitive Prompting

Instructions:
Solve the following problem by choosing and applying appropriate cognitive operations from the list below. For each step, provide your concise reasoning before moving on.

Cognitive Operations:
1. Goal Clarification: Define the objective clearly.
2. Decomposition: Break down the problem into manageable parts.
3. Filtering: Focus on the most relevant information.
4. Reorganization: Arrange the information to reveal structure
5. Pattern Recognition: Identify recurring patterns or relationships.
6. Abstraction: Extract fundamental principles from the patterns.
7. Generalization: Apply the abstracted principles to the larger problem
8. Integration: Synthesize the components into a cohesive solution.

Problem: [SPECIFIC PROBLEM TO SOLVE]

Your Response: Please start with "Goal Clarification" and proceed through each cognitive operation step by step, providing detailed reasoning and explanations.

Arithmetic Cognitive Prompting

Instructions:
Solve the following arithmetic problem by following each step of the cognitive operations listed below. For each step, provide your reasoning and calculations before moving on to the next step.

Cognitive Operations:
1. Goal Clarification: Restate the problem in your own words.
2. Decomposition: List the given information.
3. Filtering: Identify what you need to find.
4. Reorganization: Assign variables to the unknowns.
5. Pattern Recognition: define each variable clearly.
6. Abstraction: Set up equations based on the problem.
7. Generalization: Solve the equations step by step.
8. Integration: Verify your solution with the given information.

Problem: [ARITHMETIC PROBLEM TO SOLVE]

Your Response: Please start with "Restate the problem in your own words" and proceed through each cognitive operation step by step, providing detailed reasoning and calculations for each.

Fig. 1: Left: General CP, Right: CP adapted to arithmetical reasoning.

CP can be formalized as an optimization problem. Given a set of COPs $C = \{c_1, c_2, \dots, c_n\}$ and a sequence $S = \{s_1, s_2, \dots, s_k\}$ of k operations from C, the goal is to design S^* that maximizes task performance $S^* = \arg\max_{S \subseteq C} f(S)$ subject to constraints like $|S| = k$, $s_1 =$ goal clarification, and $s_k =$ integration. Here, $f(S)$ measures performance (e.g., accuracy or coherence).

This paper focuses on eight key COPs.

1. *Goal Clarification.* This operation aligns the model's reasoning with the desired outcome and minimizes distractions. All subsequent operations are guided by this goal.

2. *Decomposition:* Break the problem P into smaller sub-problems, $P_1, P_2, \ldots, P_n$. This incremental approach is particularly useful for complex, multi-step problems, such as mathematical proofs or logical reasoning. Decomposition isolates critical components for systematic problem-solving.

3. *Filtering:* Select the most relevant information from the problem set, $I_{\text{rel}} \subseteq I$. Filtering ensures the model concentrates on key details, excluding irrelevant data. By narrowing its focus, the model achieves greater accuracy and efficiency in problem-solving.

4. *Reorganization:* Rearrange data or variables to reveal patterns or simplify the problem structure. Reorganization helps the model uncover underlying relationships, making complex data more interpretable, and is particularly effective for algebraic manipulation or logical structuring.

5. *Pattern Recognition:* Identify recurring patterns or relationships, $\mathcal{P}$, that connect the problem to known solutions. Recognizing patterns accelerates problem-solving by allowing the model to apply established strategies. This enhances predictive accuracy and facilitates generalization.

6. *Abstraction:* Extract broader principles from the identified patterns, $\mathcal{P}$, for application across different problems. Abstraction helps the model transcend specific details and focus on core concepts, enabling flexible problem-solving.

7. *Generalization:* Apply the abstracted principles to solve broader problems or similar contexts. Generalization ensures that solutions are scalable and adaptable to related tasks, enhancing the model's reasoning robustness and versatility.

8. *Integration:* Synthesize the individual solutions, Q_i, into a cohesive final answer, Q, ensuring all sub-problems are resolved and producing a unified and consistent solution.

Adapting COPs to specific domains ensures that the reasoning process remains relevant and effective for each task. For arithmetic reasoning, the adapted COPs are illustrated on the right in Figure 1.

4 Cognitive Prompting Variants

CP comes in three variants. *Deterministic cognitive prompting (D-CP)* follows a fixed manual designed sequence of cognitive operations, providing structure but less adaptability. We optimized the sequence of COPs in preliminary experiments. *Self-adaptive cognitive prompting (SA-CP)* allows the model to self-select the next COP based on the task's needs, i.e., the LLM decides on its own, which COP to choose next. A prompt incorporating the following command enables self-adaptive prompting:

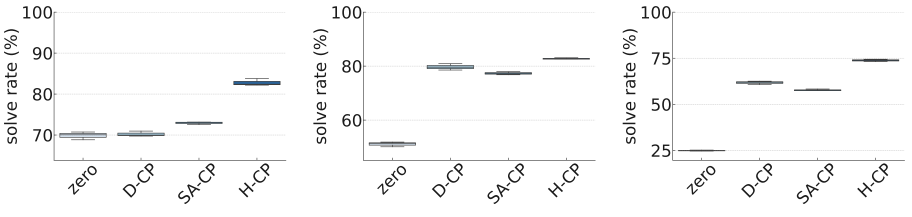
```
For each step, choose and apply the most suitable cognitive operation
from the list below and provide a concise explanation of your reasoning
before moving on to the next step.
```

This flexibility enhances problem-solving and produces more interpretable reasoning, but is based on the model's own ability to structure reasoning. *Hybrid cognitive prompting (H-CP)* uses a brief LLM-generated summary of successful problem solutions previously generated with CP and adds k summaries to the CP instruction in a hybrid CP-few-shot-CoT fashion. This variant is based on the idea two combine structured thinking with successfully solved examples, a problem-solving strategy we believe also human reasoning often follows.

5 Arithmetic Reasoning

Benchmark. We evaluate the performance of CP using Meta's LLaMA 3.1 (8B and 70B), Google's Gemma 2 (9B and 27B), and Alibaba's Qwen 2.5 (7B and 32B) models on the GSM8K dataset [2], a widely used benchmark for math problem-solving. GSM8K consists of 7,000 training and 1,500 test high-quality, grade-school math word problems, designed to assess the reasoning and mathematical capabilities of LLMs. Since CP does not require training, we exclusively evaluate performance on the test set.

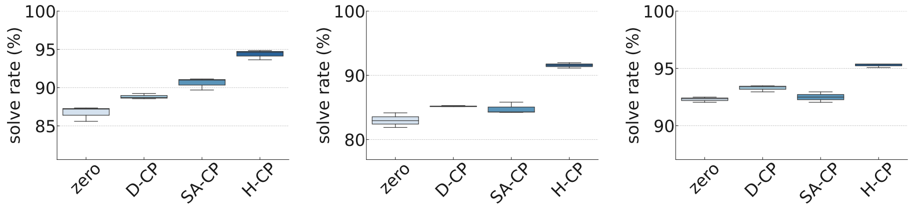
Mid-Size Models. Figure 2 presents the experimental results, comparing standard zero-shot prompting with D-CP, SA-CP, and H-CP (based on the self-adaptive CP prompt variant, $k = 2$) for the mid-size models, i.e., LLaMA 8B, Gemma 9B, and Qwen 7B. CP consistently outperforms zero-shot prompting across all variants and models, demonstrating significant improvements.



(a) LLaMA 8B (b) Gemma 9B (c) Qwen 7B

Fig. 2: Solve rates of CP strategies using mid-size models on GSM8k (3 repetitions).

Large Models. Figure 3 compares all variants across large models, including LLaMA 70B, Gemma 27B, and Qwen 32B, highlighting consistent improvements with CP. Notably, H-CP demonstrates a significant performance advantage, achieving an impressive 95% solve rate on the LLaMA 70B model. While Qwen 32B delivers excellent results even with zero-shot prompting, its performance is further enhanced by CP, particularly with the hybrid CP variant.



(a) LLaMA 70B

(b) Gemma 27B

(c) Qwen 32B

Fig. 3: Solve rates of CP strategies using large models on GSM8k (3 repetitions).

Figure 4 shows the most frequent COP sequences[1] that have automatically been chosen by SA-CP on LLaMA 70B. "GC DC PR" is the most frequent sequence, indicating its fundamental role. Shorter sequences dominate, while longer, more complex sequences are used less often. We observed similar results for the other LLMs.

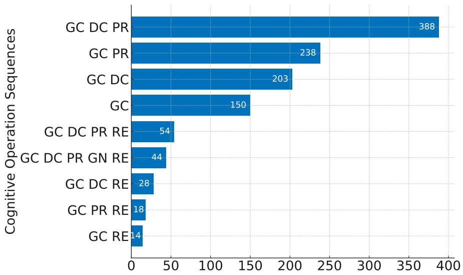


Fig. 4: SA-CP sequences, LLaMA 70B.

6 Conclusions

CP models human reasoning as a sequence of COPs delivered through structured prompts, fostering structured thinking through general or domain-specific COPs. Unlike example-based approaches like CoT, CP emphasizes high-level reasoning, making it highly adaptable across diverse tasks. Our experiments show that self-adaptive CP significantly boosts LLM performance on complex tasks, such as GSM8K math problems, with notable improvements for mid-size and larger models, though the proportional gain is greater for mid-size models. Additionally, the hybrid approach combining CoT few-shot prompting and CP delivers the best overall results across all experiments.

Our future work will focus on extending CP to additional domains and models, such as legal reasoning and strategic planning, to further validate its robustness in specialized tasks.

[1] Goal clarification (GC), decomposition (DC), pattern recognition (PR), generalization (GN), reorganization (RE)

References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Neural Information Processing Systems (NeurIPS)*, volume 35, pages 24824–24837, 2022.

[2] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[3] C. Fernando, D. Banarse, H. Michalewski, S. Osindero, and T. Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. *Neural Information Processing Systems (NeurIPS) Workshop*, 2023.

[4] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Neural Information Processing Systems (NeurIPS) Workshop*, volume 35, pages 24824–24837, 2022.

[5] C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen. Large language models as optimizers. In *International Conference on Learning Representations (ICLR)*, 2024.

[6] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Neural Information Processing Systems (NeurIPS)*, volume 36, pages 11809–11822, 2023.

[7] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

[8] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba. Large language models are human-level prompt engineers. In *International Conference on Learning Representations (ICLR)*, 2023.