Adaptive-Solver Framework for Dynamic Strategy Selection in Large Language Model Reasoning

Anonymous ACL submission

Abstract

Large Language Models (LLMs) are showcasing impressive ability in handling reasoning tasks. Humans inherently adjust problem-solving approaches based on task complexity. However, most methodologies that leverage LLMs tend to adopt a uniform approach: utilizing consistent models, prompting methods, and degrees of problem decomposition, regardless of the problem complexity. Inflexibility of these methods can bring unnecessary computational overhead or sub-optimal performance. To address this issue, we introduce an Adaptive-Solver (AS) framework that strategically adapts solving approaches to suit various problems. Given an initial solution, the framework functions with two primary modules. The initial *evaluation* module assesses the adequacy of the current solution. If improvements are needed, the subsequent *adaptation* module comes into play. Within this module, various types of adaptation strategies are employed collaboratively. Through such dynamic and multi-faceted adaptations, our framework can help reduce computational consumption or elevate performance. Experimental results from complex reasoning benchmarks reveal that instantiation methods developed based on the AS framework can significantly reduce API costs (up to 62%) while maintaining superior performance, or enhance performance across all tasks.[1]

1 Introduction

Large Language Models (LLMs) exhibit remarkable proficiency across various reasoning tasks. However, while the potential of LLMs in addressing intricate problems is undeniable, the quest to identify the most effective problem-solving strategy to maximize their performance remains largely untapped. To tackle this problem, we turn to draw inspiration from the innate problem-solving approaches employed by humans. The human cognitive framework consists of two distinct systems: *System 1* for intuitive thinking, and *System 2* for deeper, analytical reasoning (Sloman, 1996; Daniel, 2017). These systems are utilized dynamically and adaptably, catering to a range of problem complexities, thereby ensuring both efficiency and accuracy in problem-solving.

Likewise, when faced with complex challenges, humans often break down the problem into more detailed sub-questions, ensuring a lucid formulation of the task. For simpler question, a direct, singular line of reasoning is typically employed. If their initial solution not meet expectations, humans naturally pivot their approach in pursuit of a more effective resolution. Recognizing the multifaceted nature of real-world challenges and drawing inspiration from human problem-solving methodologies, it stands to reason that machines too should be equipped with the capacity to dynamically adjust their problem-solving strategies. This adaptation might encompass variations in the underlying LLM models, sample size, granularity in problem-decomposition, or prompting methods.

Current research trends often employ a static solver[2], neglecting the distinct characteristics of individual problems. This inflexibility in adjusting the solver to diverse problems can result in unnecessary resource consumption and sub-optimal performance. For example, GPT-4, while possessing remarkable capabilities, comes with a significant API cost. Utilizing a more cost-effective model for simpler queries can be a strategy to reduce expenses. Additionally, at the problem-solving method layer, the Chain-of-thought (CoT) (Wei et al., 2022) prompts LLMs to generate an intermediary reasoning process to yield reliable results. However, its reliance on a single-turn of reason-

[1] We will release all our code upon acceptance to facilitate research on this line.

[2] In this context, a solver encompasses all elements integral to problem-solving, including the LLM model, prompting techniques, decomposition strategies, and so forth.

ing, without explicit sub-problem decomposition, makes it less suitable for complex challenges. To enhance CoT, Self-consistency (SC) (Wang et al., 2023c) generates results multiple times and selects the answer through majority voting, while Least-to-most (L2M) decomposes the main problem into distinct sub-problems. Despite improved performance, these methods face limitations: they either use a fixed sampling quantity or lack the flexibility to adjust the granularity of problem decomposition—such as modifying the number of sub-problems—based on the problem's complexity. A decomposition that is too coarse may oversimplify the main question, while an excessively detailed breakdown can increase the risk of decomposition errors. Balancing granularity is essential to optimize problem-solving effectiveness. Thus, we argue that distinct problems necessitate dynamically customized solvers to achieve both optimal cost-efficiency and enhanced performance.

In response to the clear demand for dynamic problem-solving methods, we propose the Adaptive-Solver (AS) framework. The AS framework is structured around two core modules: the *evaluation* module and the *adaptation* module. The *evaluation* module assesses the current solution's efficacy, determining whether it meets the problem-solving standards. Should the solution not meet the requisite quality, the *adaptation* module is triggered, adjusting the solving strategy for the following phase. Within the *adaptation* module, four adaptation strategies are devised: (1) *Model Adaptation*: Shifting to a more powerful, albeit resource-intensive, LLM when necessary; (2) *Sample Size Adaptation*: Initializing the sample size with small value and incrementally lifting it when needed; (3) *Prompting Method Adaptation*: Varying the prompting techniques to better align with the complexity of the problem; (4) *Decomposition Granularity Adaptation*: Modulating the granularity of problem decomposition according to the problem complexity. These adaptation strategies can be combined to achieve a dynamic and multifaceted adjustment to the current solving approach.

Based on the AS framework, two instantiation methods—AS-MS and AS-PD—are proposed, by integrating different adaptation strategies. AS-MS focuses on reducing computational consumption through combining *model adaptation* and *sample size adaptation*. AS-PD aims at improving performance by integrating *prompting method adaptation* and *decomposition granularity adaptation*.

Extensive experiments across 8 reasoning tasks corroborate the effectiveness of the Adaptive-Solver and draw several crucial findings: (1) The AS-PD method consistently elevates performance across every task. This underscores the merit of dynamic strategy selection in enabling LLMs to select the optimal reasoning technique for multi-faceted challenges. (2) The AS-MS method notably reduces API cost (up to 62%), while upholding a superior performance.

Our contributions can be distilled into the following key points: (1) We introduce the Adaptive-Solver framework. It is adept at strategically selecting the optimal solving methodologies tailored to the intrinsic characteristics of a given problem. (2) We propose four versatile adaptation strategies concerning model selection, sample size, prompting methods, and decomposition granularity. (3) We devise two instantiation methods to respectively reduce computational consumption and enhance overall performance. (4) Experiments underscore the superiority of the Adaptive-Solver framework, demonstrating marked enhancements in computational efficiency and performance outcomes.

2 The Adaptive-Solver Framework and Its Instantiations

Overview. Our Adaptive-Solver (AS) framework integrates multiple solvers and dynamically determine the most suitable solver according to the problem characteristic. This framework comprises two main modules: *evaluation* module and *adaptation* module. The framework's workflow is depicted in Figure 1(a): 1) Given a problem, candidate solution is generated by the current solver. The *evaluation* module assesses whether the solution successfully meets the evaluation criteria. If the criteria are satisfied or the maximum predefined number of solving attempts is reached, the solving process terminates. 2) If the criteria are not met, the *adaptation* module will be activated to adjust the solver, and then the process proceeds to the next solving round by executing 1) again. The *adaptation* module activates solvers sequentially in a pipeline, specifying the employed solvers and their order. The configuration of this pipeline is automatically determined for each dataset, as explained in § 2.3. Within this module, four key adaptation strategies are designed to provide guidance on how to adjust the solver. *Model Adaptation* (shown in Figure 1(a1)): Switching to a more advanced LLM to

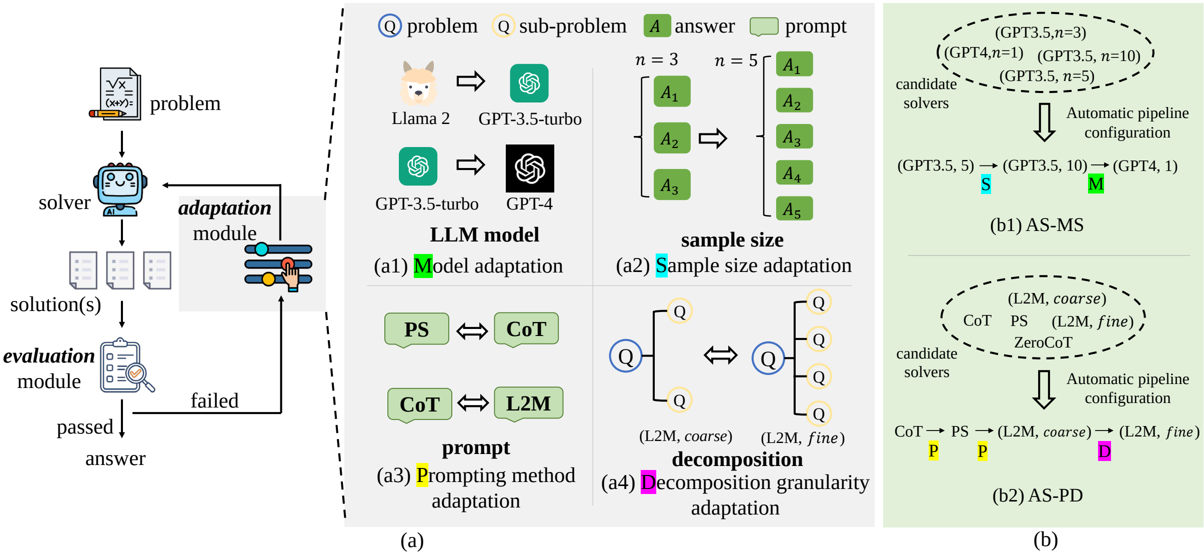


Figure 1: The Adaptive-Solver (AS) framework and its four adaptation strategies are shown in (a), while the two instantiation methods of the framework are depicted in (b).

ensure the accuracy of solving complex problems. *Sample Size Adaptation* (shown in Figure 1(a2)): Gradually increasing the sample size within the self-consistency strategy to enhance the likelihood of correctly solving problems. *Prompting Method Adaptation* (shown in Figure 1(a3)): Alternating between different prompting techniques to suit the problem's characteristic. *Decomposition Granularity Adaptation* (shown in Figure 1(a4)): Fine-tuning the level of decomposition granularity to achieve the most effective granularity for addressing problems of different complexities. Based on the AS framework, we introduce two instantiation methods: AS-PD aims to enhance performance and AS-MS focuses on reducing costs (shown in Figure 1(b1) and (b2)).

2.1 Evaluation Module

The *evaluation* module aims to evaluate whether the current solver is sufficient to resolve the problem, and decide when to adapt the solver. This paper adopts a self-consistency-based metric to evaluate the answer. (Wang et al., 2023c) found that the *consistency* (in terms of % of decodes agreeing with the final aggregated answer) is highly correlated with accuracy. This enables us to leverage *consistency* to estimate the likelihood of the current answer being correct and reflect the confidence of model prediction. Therefore, in our implementation of the proposed framework, each solver samples N diverse solutions during a single solving round and then the metric *consistency* is calculated. If the *consistency* (i.e., # of the most consistent answer / N) reaches a predefined threshold θ, the solving process terminates. In this paper, unless otherwise specified, the default value of N and θ are 3 and 1.0.

2.2 Adaptation Module

The *adaptation* module aims to address the shortcomings of the "one solver for all problems" strategy. It dynamically adapt the solver to different problems. This enables it to identify an appropriate solver for each problem, one that helps reduce computational costs or enhance performance. We adopt a straightforward approach to implement the adjustment of solvers, which involves determining a list of solvers and switching to the next solver in the list from the current one when adaptation is needed. Therefore, we denote an adaptation strategy as a list $\mathcal{A}$ of solvers:

$$\mathcal{A} = [\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_n], \mathcal{S}_n = (m_n, s_n, p_n, d_n, ...)$$

where $\mathcal{S}_n$ represents the n-th solver, and each solver $\mathcal{S}_n$ can be represented as a tuple of elements such as LLM model denoted as m_n, sample size denoted as s_n, prompting method denoted as p_n, and decomposition granularity denoted as d_n. For simplicity, we represent $\mathcal{S}_n$ with only the adjustable elements. For example, the *model adaptation* strategy is represented as $[m_1, m_2, ..., m_n]$. We propose a method to automatically determine the solver list given a dataset, as introduced in section § 2.3.

Given the list $\mathcal{A}$, the solvers are sequentially activated until either all solvers are tried or the evaluation criteria are satisfied. If the criteria are met, the current solver is chosen. If none of the solvers meet the criteria, two strategies are available to select the final solver: 1) Choose the last solver in the list. 2) Select the solver with the highest *consistency*, as calculated in the *evaluation* module. In the case of multiple solvers having the same highest *consistency*, the most recently invoked solver is selected. We use the first strategy for the *model adaptation* and *sample size adaptation*, and the second strategy for the *prompting method adaptation* and *decomposition granularity adaptation*.

Model adaptation: *Model adaptation* is denoted as $[m_1, m_2, ..., m_n]$, where m_n represents a stronger but more expensive LLM than m_{n-1} does. As illustrated in Figure 1(a1), an example can be [GPT-3.5, GPT-4].

Sample size adaptation: *Sample size adaptation* is denoted as $[s_1, s_2, ..., s_n]$, where s_n represents the number of sampled answers and $s_n > s_{n-1}$. Furthermore, we lower the threshold θ (§ 2.1) in $evaluation$ module when the sample size gets bigger. As illustrated in Figure 1(a2), an example of *sample size adaptation* can be [3, 5] and the corresponding thresholds can be [1.0, 0.8].

Prompting method adaptation: *Prompting method adaptation* is denoted as $[p_1, p_2, ..., p_n]$, where p_n represents prompting methods such as CoT, L2M. As illustrated in Figure 1(a3), an example can be [CoT, L2M] or [L2M, CoT].

Decomposition granularity adaptation: To mitigate the constraint posed by L2M's inflexibility in adapting problem decomposition granularity, we introduce an adaptation method that tailors the decomposition granularity to each specific problem. We design three different variants of L2M prompt, denoted as (L2M, *coarse*), (L2M, *medium*) and (L2M, *fine*), where the decomposition granularity ranges from coarser to finer. We illustrate how to construct L2M's variants in Appendix A.6. The only difference among them is the decomposition granularity in their demonstrations, as shown in A.8.5. *Decomposition granularity adaptation* is denoted as $[d_1, d_2, ..., d_n]$, where $d_n \in \{coarse, medium, fine\}$. In this paper, the modulation of decomposition granularity is solely available when using the prompting method L2M. As shown in Figure 1(a4), an example can be [*coarse*, *fine*] or [*fine*, *coarse*].

2.3 Two Instantiations and Automatic Pipeline Configuration

The pipeline configuration's objective is to choose from a set of candidate solvers and determine their sorting order. To determine the optimal list of solvers for each dataset, a subset is sampled as a validation set for the search process.

AS-MS and its pipeline configuration algorithm. The pipeline configuration of AS-M is depicted in Figure 1(b1). The input comprises a set of solvers, wherein the variable elements include the LLM model and sample size. The output yields a pipeline of solvers aimed at minimizing costs on the validation set, while ensuring that performance decrease remains within an acceptable range. Refer to Algorithm 1 in Appendix A.7 for more details.

AS-PD and its pipeline configuration algorithm. The pipeline configuration of AS-PD is shown in Figure 1(b2). The input comprises a set of solvers, wherein the prompting method and decomposition granularity are variable elements. The output yields a pipeline of solvers aimed at maximizing overall accuracy on the validation set. Refer to Algorithm 2 in Appendix A.7 for more details.

3 Experimental Setup

3.1 Datasets

The proposed method is evaluated on 8 datasets from three categories of reasoning tasks. **Arithmetic Reasoning**: GSM8K (Cobbe et al., 2021a), SVAMP (Patel et al., 2021), AQuA (Ling et al., 2017), AddSub (Hosseini et al., 2014), SingleEq (Koncel-Kedziorski et al., 2015) and MultiArith (Roy and Roth, 2015); **Commonsense Reasoning**: CSQA (Talmor et al., 2019); **Symbolic Reasoning**: Last Letter Concatenation (LLC) (Wei et al., 2022). We partition each dataset into a validation set and a test set. The validation set is utilized to identify the optimal solver list for our method, while the test set is employed to compare the performance and cost of all methods. Refer to Appendix A.3 for more dataset details.

3.2 Baselines

1) For AS-PD, the baselines are various prompting methods. We include two types of prompting baselines: single-solution promptings solve problems in a single-turn, including ZeroCoT (Kojima et al., 2022), PS (Wang et al., 2023a), CoT (Wei et al., 2022) and L2M (Zhou et al., 2023); multi-solution promptings solve problems for multiple

times, including CoT_SC (Wang et al., 2023c) and PHP (Zheng et al., 2023). We use GPT-3.5-turbo as model for all these prompting methods. 2) For AS-MS, the baselines are the methods that using only weaker or stronger LLM, i.e., GPT-3.5-turbo or GPT-4 in this paper. We uniformly employ ZeroCoT as the prompting method for all LLMs. See implementation details in Appendix A.4.

4 Experimental Results

4.1 Main Results of AS-PD

Results on Arithmetic Reasoning. Table 1 reports the accuracy comparison of our method AS-PD with existing single-solution and multi-solution methods on the arithmetic reasoning datasets.

1) Adaptive-Solver framework effectively enhances LLM's performance. Our method, AS-PD, consistently surpasses all baseline methods across various arithmetic datasets. Specifically, in terms of average accuracy, AS-PD achieves 89.3%, outperforming the best result among the baselines (85.8%) by 3.5%. This outcome highlights the efficacy of our approach in enhancing the reasoning capabilities of LLMs by dynamically selecting the most suitable prompting method and decomposition granularity.

2) Multi-solution methods outperform single-solution approaches in overall performance. Specifically, CoT_SC is designed to produce multiple solutions within a single solving round, while PHP adopts a multi-round solving approach but yields a single solution in each round. Our proposed methods integrate both the paradigms, enabling iterative problem-solving across multiple rounds, with generating multiple solutions at each iteration. All of these methods exhibit enhancements over single-solution prompting techniques, with our approaches delivering the most notable performance gains. These findings underscore the effectiveness of multi-solution promptings in significantly improving the model's accuracy.

3) Different prompting has its own strengths and weaknesses. ZeroCoT and CoT represent methods that solve problems in a single stage without prior explicit planning or problem decomposition. PS and L2M represent methods that address problems in a two-stage approach, commencing with explicit planning or problem decomposition. ZeroCoT outperforms PS on two datasets while underperforming on the remaining four datasets. Similarly, CoT outperforms L2M on four datasets but doesn't do as well on the other two. This indicates that each of them has its own strengths and weaknesses, making them suitable for different types of problems.

Results on Commonsense and Symbolic Reasoning. Due to commonsense reasoning problems typically do not entail multi-step solving or problem decomposition, it becomes unnatural to apply L2M in this context. Besides, decomposition granularity adaptation is also unnecessary to the task of last letters concatenation. Therefore, we only use *prompting method adaptation* strategy in these two scenarios and our method is denoted as AS-P. Table 2 reports the results on the commonsense reasoning dataset CSQA and the symbolic reasoning dataset LLC. We observe that our method consistently outperforms all baselines. Specifically, AS-P surpasses the best baseline on CSQA and LLC by 2.1% and 0.8%, respectively.

4.2 Main Results of AS-MS

The primary goal of AS-MS is to cut down on expensive API calls or computational resources required to solve a problem, while maintaining performance. We validate the effectiveness of AS-MS by examining both the performance and cost. Table 3 presents the performance and cost comparison of AS-MS with the baselines that use only single LLM model or fixed sample size in SC.

AS-MS reduces the overall API cost while maintaining superior performance. From Table 3, we can observe that: 1) GPT4 surpasses GPT3.5 even with sample size up to 10, by a significant margin. Specifically, GPT4 leads GPT3.5 (SC=3) by approximately 6-16%. However, this performance improvement is accompanied by a relatively higher cost, roughly 7-13 times expensive. 2) AS-MS performs at a comparable level to GPT4, and in certain cases, it even outperforms it slightly. Moreover, this combination significantly reduces the overall API cost, saving approximately 46-62% of API cost compared to the case using GPT4 alone.

4.3 Efficiency Analysis

We evaluate our methods' time efficiency by calculating the average solving rounds. The results are demonstrated in Table 9 in Appendix A.5. We can observe that despite our methods employ a multi-round solving strategy, there is no notable increase observed in the average number of iterations. The increase is basically around 1.5 times

Table 1: Performance comparison of AS-PD with baselines on the arithmetic reasoning datasets. SS: Single-solution prompting, MS: Multi-solution prompting. The best results are boldfaced. The solver list used by AS-PD on each dataset are as follows: GSM8K: [ZeroCOT, COT, (L2M, *coarse*), PS, (L2M, *medium*)], SVAMP: [PS, ZeroCOT, (L2M, *fine*), (L2M, *coarse*), (L2M, *medium*)], AQuA: [ZeroCOT, COT, (L2M, *coarse*), PS, (L2M, *medium*)], AddSub: [COT, (L2M, *fine*)], SingleEq: [COT, (L2M, *coarse*), PS], MultiArith: [COT, L2M, (L2M, *fine*)]. The other elements are unified as: LLM model: GPT-3.5-turbo-0301, sample size: 3, threshold θ: 1.0.

Type	Method	GSM8K	SVAMP	AQuA	AddSub	SingleEq	MultiArith	Average
SS	ZeroCoT	79.6	79.1	55.5	81.0	89.6	96.3	80.2
	PS	78.8	80.0	59.8	87.3	93.9	96.0	82.6
	CoT	80.8	80.6	57.5	88.9	96.1	98.3	83.7
	L2M	77.7	83.1	52.4	90.5	93.3	93.3	81.7
MS	CoT_SC	84.3	82.2	63.4	90.6	96.3	97.8	85.8
	PHP	85.5	81.4	63.6	86.4	92.8	98.2	84.7
	AS-PD	**89.6**	**90.0**	**68.1**	**92.2**	**97.0**	**98.8**	**89.3**

Table 2: Performance on CSQA and LLC datasets. The solver list of AS-P on each dataset: CSQA: [ZeroCoT, PS, CoT], LLC: [L2M, CoT]

Method	CSQA	LLC
ZeroCoT	70.4	71.6
PS	69.8	63.6
CoT	73.1	92.6
L2M	-	95.0
CoT_SC	72.1	92.6
AS-P	**75.3**	**95.8**

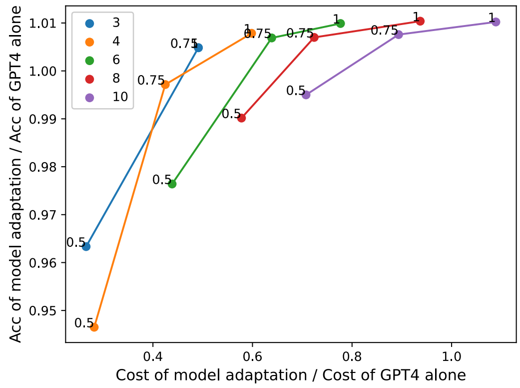


Figure 2: The balance between performance and cost in *model adaptation*. The horizontal axis shows the cost ratio of *model adaptation* to using GPT4 alone, while the vertical axis represents the accuracy ratio. Each color corresponds to a sample size N, and each point is labeled with a threshold θ.

that of single-round solving methods. This can be attributed to the fact that the initial solver resolves the majority of problems, with subsequent solvers being invoked only in a few necessary cases.

4.4 Analysis of Model Adaptation

We investigate how the cost and performance change when we modify the criteria of the *evaluation* module, achieved by adjusting both the sample size N and the threshold θ. By focusing on *model adaptation*, we simplify the solver list as [(GPT3.5, n, θ), (GPT4, 1)], and explore the effect of sample size n and threshold θ on its performance and cost.

There exists a trade-off between the cost and performance in *model adaptation*. The visualization in Figure 2 illustrates the relationship between the cost and the performance of *model adaptation*. The performance increases as the cost rises, gradually reaching convergence. This indicates a trade-off between the cost and the performance in *model adaptation*. When we pick an appropriate sample size and threshold, we can attain commendable performance at a comparatively modest expense.

4.5 Analysis of Decomposition Granularity Adaptation

We delve deeper into examining the efficacy of adapting decomposition granularity. This is achieved through a comparative analysis of our method against its variants that fix decomposition granularity. In order to eliminate the impact of a multi-round solving strategy, we permit all non-adaptive decomposition prompts to address problems across multiple rounds, with a maximum limit of 3 rounds. The results are reported in Table 4. The *decomposition granularity adaptation* method AS-D consistently outperforms the non-adaptive baseline on all datasets. On average, AS-D surpass [L, L, L] by 1.5%. This indicates that *decomposi-*

Table 3: Performance and cost are compared between AS-MS and baseline methods. On GSM8K, SVAMP and LLC, the solver list of AS-MS is [(GPT3.5, n=3, θ=1.0), (GPT3.5, n=5, θ=0.8), (GPT3.5, n=10, θ=0.6), (GPT4, n=1)]. On CSQA, the optimal solver list is [(GPT3.5, n=3, θ=1.0), (GPT4, n=1)]. n is sample size. We employ ZeroCoT as the prompting method across all approaches.

LLM Model	n	GSM8K		SVAMP		CSQA		LLC	
		ACC	Cost ($)	ACC	Cost ($)	ACC	Cost ($)	ACC	Cost ($)
GPT3.5	3	84.9	2.9395	82.5	1.4215	72.6	1.3798	75	0.2608
	5	86.9	4.7271	86.3	2.3005	73.9	2.1995	87.8	0.4043
	10	88.5	9.4168	86.9	4.5591	73.8	4.3416	89	0.7955
GPT4	1	93.5	25.1662	88.8	11.1847	80.7	11.3549	91.8	3.2407
AS-MS	-	92.0	9.6822	88.9	4.2251	79.1	6.0527	93.8	1.2305
Acc. Gain / Saved $	-	-1.5 / 61.5%		+0.1 / 62.2%		-1.6 / 46.7%		+2 / 62.0%	

Table 4: Ablation experiment investigating the efficacy of *decomposition granularity adaptation*. Let L = L2M, L1 = (L2M, *coarse*), L2 = (L2M, *medium*), L3 = (L2M, *fine*). The solver list of AS-D on each dataset are as follows: GSM8K, AQuA and AddSub: [L1, L2, L3], SVAMP: [L3, L1, L2], SingleEq and MultiArith: [L1, L3].

Method	GSM8K	SVAMP	AQuA	AddSub	SingleEq	MultiArith	Average
[L, L, L]	86.1	87.0	61.9	92.4	95.3	96.3	86.5
AS-D	**87.5**	**89.0**	**63.3**	**92.9**	**96.1**	**99.2**	**88.0**
Acc. Gain	+1.4	+2.0	+1.4	+0.5	+0.8	+2.9	+1.5

tion granularity adaptation can enhance the performance by dynamically adjusting the decomposition granularity for each problem.

4.6 Analysis of Prompting Method Adaptation

To further investigate the efficacy of *prompting method adaptation*, we compare our method with its non-adaptive variants that do not adjust promptings. The results are reported in Table 5. L2M is not suitable for CSQA, so we do not report the result. 1) The optimal prompting method varies depending on the dataset. For instance, on GSM8K [Z, Z, Z, Z] performs better than [P, P, P, P] but exhibits lower performance on the SVAMP dataset. 2) The adaptive method AS-P outperforms the non-adaptive variants on alomost all datasets. This suggests that *prompting method adaptation* can dynamically select a suitable prompting method for different problems across various datasets, resulting in an improved performance.

5 Related Work

Reasoning with LLM prompting. It is widely recognized that complex reasoning problems are quite challenging for language models. Such problems include mathematical reasoning (Lu et al., 2023; Cobbe et al., 2021b), commonsense reasoning (Talmor et al., 2018), symbolic reasoning (Wei et al., 2022) and logical reasoning (Creswell et al., 2023). The recently proposed CoT (Wei et al., 2022) prompting significantly enhances the complex reasoning capabilities of LLMs, by generating intermediate reasoning steps to obtain the answer. Similarly, (Kojima et al., 2022) proposes Zero-CoT to elicit reasoning step generation without exemplars. PAL (Gao et al., 2023) and PoT (Chen et al., 2022) generate programs to represent the reasoning process and utilize a code interpreter to execute the programs. CoT has inspired diverse prompting methods aimed at further enhancing the complex reasoning capabilities of LLMs. Among these works, there are two prevailing technical approaches. The first type of methods adopt the idea of "divide and conquer". PS prompting (Wang et al., 2023a) devises a plan to divide the entire task into smaller subtasks, and then carry out the subtasks according to the plan. Besides, some methods (Zhou et al., 2023; Khot et al., 2023) decompose the main problem into simpler sub-problems to solve. The second type of methods adopt the idea of "try more". SC (Wang et al., 2023c) decoding strategy improves CoT by sampling multiple solutions in a single round and determining the final answer through majority voting. PHP (Zheng et al., 2023) solves problems iteratively over multi-

Table 5: Ablation study of *prompting method adaptation*. Let Z = ZeroCoT, P = PS, C = CoT, L = L2M. The solver list of AS-P on each dataset are as follows: GSM8K: [Z, C, P, L], SVAMP: [L, P, C, Z], AQuA: [P, C, Z, L], MultiArith: [C, Z, P, L], CSQA: [Z, P, C], LLC: [L, C].

Method	GSM8K	SVAMP	AQuA	MultiArith	CSQA	LLC
[Z, Z, Z, Z]	88.5	86.5	67.6	97.4	**75.2**	83.3
[P, P, P, P]	87.8	88.7	67.6	98.0	73.8	75.5
[C, C, C, C]	88.1	84.8	65.2	98.8	73.5	92.5
[L, L, L, L]	86.5	87.9	60.8	97.0	-	95.3
AS-P	**89.5**	**88.9**	**68.6**	**99.2**	75.0	**95.8**
Acc. Gain	+1.0	+0.2	+1.0	+0.4	-0.2	+0.5

ple rounds and utilizes generated answers as hint to guide the subsequent responses. Besides, some works (Yao et al., 2023; Xie et al., 2023) sample multiple responses for each step and integrate step-wise self-evaluation to guide the generation of a whole solution. However, most of the existing works construct a fixed solver for different problems, regardless of their varied complexity, which may result in unnecessary computational overhead or sub-optimal performance. Some efforts have been made to improve computational efficiency. (Chen et al., 2023) and (Yue et al., 2023) cascade weaker LLMs and stronger LLMs to reduce overall costs while maintaining superior performance. (Aggarwal et al., 2023) dynamically adjusts the number of samples in Self-Consistency (Wang et al., 2023c) based on a stopping criterion, in order to minimize the sample budget. However, these approaches focus on adjusting a single dimension, either LLM model or sample size. In contrast, our proposal introduces a comprehensive framework capable of adapting a solver from various perspectives, such as the LLM model, sample size, prompting method, and decomposition granularity. This flexibility allows for the implementation of different adaptation strategies within our framework. Moreover, these adaptation strategies can be combined in a flexible manner to create diverse instantiations.

Automated feedback for LLMs. Another relevant research area is providing automated feedback to the LLM's response. (Pan et al., 2023) divide automated feedback into two types according to the sources: self-feedback and external feedback. Self-feedback denotes the feedback originated from the LLM itself, i.e, self-evaluation (Madaan et al., 2023; Weng et al., 2023; He et al., 2022). External feedback represents the feedback derived from external models (Wang et al., 2023b), tools (Gou et al., 2023), metrics (Jung et al., 2022) and knowledge bases (Yu et al., 2023). The *evaluation* module in our framework can be implemented based on various automated feedback methods. Since we focus on the *adaptation* module, for simplicity, we adopt a self-consistency-based metric (i.e., *consistency*) (Wang et al., 2023c) to evaluate the answer.

6 Conclusion and Discussion

We propose the Adaptive-Solver framework, designed to dynamically tailor solving strategies for LLMs across diverse reasoning scenarios. Central to this framework are two modules: the initial evaluation module, which assesses the adequacy of a given solution, and the subsequent adaptation module if refinement is necessary. Herein, three adaptation strategies are leveraged: *model adaptation, prompting method adaptation*, and *decomposition granularity adaptation*. Utilizing the framework, we introduce two instantiation methods—AS-PD and AS-MS—aimed at enhancing performance and reducing costs, respectively. Our experimental results highlight the effectiveness of this framework. Specifically, AS-PD consistently enhances the ability of the LLM by identifying optimal prompting methods and decomposition granularity. Notably, AS-MS achieves a significant reduction in API costs, cutting them by up to 62%, while maintaining or even amplifying performance. This framework propels us into a promising direction in dynamic strategy selection for LLMs. Viewing from a higher point, every solver – be it model, prompting, decomposition, or augmented tools – can be regarded as a potential candidate in the component pool. The LLMs, armed with this framework, exhibit the flexibility to dynamically compose selected candidates, paving the way to optimal solution paths.

7 Limitations

There are some limitations within this work that can be addressed in future research.

1) The consistency check employed in the *evaluation* module requires multiple samplings of answers, which incurs more computational resource. Additionally, its classification accuracy may be constrained by its inherent nature. To enhance the *evaluation* module, one could investigate additional techniques, such as LLM-based self-evaluation and fine-tuned models for automated evaluation. These approaches can leverage the text information in the LLM output without requiring multiple answers.

2) Our pipeline configuration algorithm customizes the solver list for each dataset rather than for each specific problem. In future research, we aim to investigate the customization of solver list for individual problems, with the potential to further enhance overall performance.

8 Ethics

We used 8 public datasets, among which GSM8K and SVAMP use the MIT License code, AQUA uses the Apache-2.0 code, the remaining datasets are unspecified. The suggested prompts do not gather or utilize personal information regarding others. The prompts employed are enumerated in the Appendix. None of the prompts employed in this study include words that discriminate against any person or group. The prompts in this research are designed not to adversely affect the safety of others.

References

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam . 2023. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with LLMs. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021a. Training verifiers to solve math word problems.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021b. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2023. Selection-inference: Exploiting large language models for interpretable logical reasoning. In *The Eleventh International Conference on Learning Representations*.

Kahneman Daniel. 2017. *Thinking, fast and slow*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.

Hangfeng He, Hongming Zhang, and Dan Roth. 2022. Rethinking with retrieval: Faithful large language model inference. *arXiv preprint arXiv:2301.00303*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, Doha, Qatar. Association for Computational Linguistics.

Jaehun Jung, Lianhui Qin, Sean Welleck, Faeze Brahman, Chandra Bhagavatula, Ronan Le Bras, and Yejin Choi. 2022. Maieutic prompting: Logically consistent reasoning with recursive explanations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1266–1279, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang.

2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

Pan Lu, Liang Qiu, Wenhao Yu, Sean Welleck, and Kai-Wei Chang. 2023. A survey of deep learning for mathematical reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14605–14631, Toronto, Canada. Association for Computational Linguistics.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2023. Automatically correcting large language models: Surveying the landscape of diverse self-correction strategies. *arXiv preprint arXiv:2308.03188*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.

Steven A Sloman. 1996. The empirical case for two systems of reasoning. *Psychological bulletin*, 119(1):3.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634, Toronto, Canada. Association for Computational Linguistics.

Tianlu Wang, Ping Yu, Xiaoqing Ellen Tan, Sean O'Brien, Ramakanth Pasunuru, Jane Dwivedi-Yu, Olga Golovneva, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023b. Shepherd: A critic for language model generation. *arXiv preprint arXiv:2308.04592*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023c. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. 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*.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. *CoRR, abs/2212.09561*.

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, Xu Zhao, Min-Yen Kan, Junxian He, and Qizhe Xie. 2023. Decomposition enhances reasoning via self-evaluation guided decoding. *arXiv preprint arXiv:2305.00633*.

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

Wenhao Yu, Zhihan Zhang, Zhenwen Liang, Meng Jiang, and Ashish Sabharwal. 2023. Improving language models via plug-and-play retrieval feedback. *arXiv preprint arXiv:2305.14002*.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. 2023. Large language model cascades with mixture of thoughts representations for cost-efficient reasoning. *arXiv preprint arXiv:2310.03094*.

Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. 2023. Progressive-hint prompting improves reasoning in large language models. *arXiv preprint arXiv:2304.09797*.

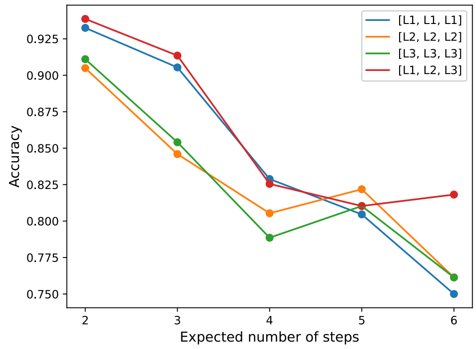
Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

A Appendix

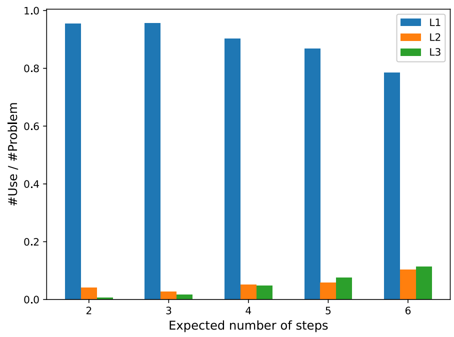
A.1 Additional analysis of decomposition granularity adaptation

Our method of constructing L2M's variants can indeed control the granularity in decomposition. Table 6 demonstrates the average number of sub-problems obtained by using L2M and L2M's variants. We observe that finer-grained decomposition prompt indeed leads to a greater number of sub-problems on average on the same dataset. This validates the effectiveness of controlling the granularity in the actual problem decomposition by modulating the granularity in the exemplars.

Decomposition granularity adaptation **tailors decomposition granularity to problems with varied difficulties.** We conduct a performance comparison of different decomposition prompts when faced with increasingly challenging problems. The visualization is presented in Figure 3. 1) In Figure 3(a), it is evident that using a fixed-granularity decomposition method does not guarantee optimal performance across all problems. For example, for problems requiring fewer than 5 steps, the coarser-grained decomposition [L_1, L_1, L_1] performs better than [L_2, L_2, L_2]. However, as the difficulty continues to increase, the finer-grained method [L_2, L_2, L_2] exhibits superior performance. This demonstrates that problems of varying difficulty require different levels of decomposition. However, our adaptive decomposition method, denotes as [L_1, L_2, L_3], consistently perform well across all settings, showcasing the advantage of *decomposition granularity adaptation*. 2) Figure 3(b) offers a further elucidation of the superior performance achieved by the adaptive method. The method dynamically selects the decomposition prompts for various problems. As the complexity of the problem escalates, it progressively enhances the utilization of finely-grained decomposition prompts, thereby resulting in an enhancement of overall performance.



(a) Accuracy varies with the problem difficulty.



(b) The usage ratio of different decomposition prompts varies with the problem difficulty.

Figure 3: Analysis of decomposition granularity adaptation on GSM8K. The problem difficulty is measured by the number of expected solving steps, provided by the GSM8K dataset. L_1 = (L2M, *coarse*), L_1 = (L2M, *medium*), L_3 = (L2M, *fine*).

A.2 Additional analysis of prompting method adaptation

Prompting method adaptation **combines the advantages of different prompting methods.** We use the simplified implementation of *prompting method adaptation* [(CoT, $n = 3$), (L2M, $n = 3$)] (denoted as [CoT*, L2M*]) to study how it works. As presented in Table 7 (Appendix A.2), we categorize all problems into four distinct groups based on the individual performance of CoT* and L2M*. We then measure the accuracy of the adaptive method [CoT*, L2M*] on each group, as well as the frequency of using CoT* and L2M* within the adaptive method. For the problems that both CoT* and L2M* successfully solve, we observed that [CoT*, L2M*] basically yields correct answers. Furthermore, for the subset of problems where either CoT* or L2M* succeeds while the other does not, [CoT*, L2M*] effectively address the majority (60%-70%) of them. These findings indicate that the adaptive approach effectively harnesses the complementary strengths of both prompting methods, leading to improved performance.

A.3 Dataset Details

In Table 8, you can find the dataset statistics.

Table 6: Average number of sub-problems of various decomposition prompting methods.

Method	GSM8K	SVAMP	MultiArith	AddSub	SingleEq	AQuA	Average
L2M	3.61	2.76	2.80	2.51	2.63	3.08	2.90
(L2M, *coarse*)	2.60	1.88	2.06	1.73	1.77	2.19	2.04
(L2M, *medium*)	3.6	2.76	2.73	2.44	2.54	2.74	2.80
(L2M, *fine*)	4.46	3.56	3.51	2.85	3.15	3.57	3.52

Table 7: Analysis of prompting method adaptation. CoT* ✓and L2M* ✗ means the problems that CoT* solves successfully while L2M* fails. CoT* and L2M* are respectively the self-consistency version of CoT and L2M.

Dataset	CoT*	L2M*	# problems	# correct problems by [CoT*, L2M*]	CoT* usage count	L2M* usage count
GSM8K	✓	✓	995	984 (98.9%)	884	11
	✓	✗	123	76 (61.8%)	62 (50.4%)	61
	✗	✓	84	56 (66.7%)	31	53 (63.1%)
	✗	✗	117	25 (21.4%)	50	67
SVAMP	✓	✓	762	755 (99.1%)	692	70
	✓	✗	51	34 (66.7%)	31 (60.8%)	20
	✗	✓	94	65 (69.1%)	34	60 (63.8%)
	✗	✗	93	15 (16.1%)	45	48

A.4 Implementation Details

We use GPT-3.5-turbo-0301 as the LLM model unless otherwise stated. We set the temperature as 0 for the greedy decoding strategy used in single-solution baselines and PHP, while 0.7 for the methods with self-consistency strategy. When using self-consistency, we set the sampling size to 3 unless otherwise specified. Our experiments were conducted from June 2023 to September 2023. During this period, the API prices for GPT-3.5-turbo were {"input": $0.0015 / 1K tokens, "output": $0.002 / 1K tokens}, while for GPT-4, the corresponding rates were {"input": $0.03 / 1K tokens, "output": $0.06 / 1K tokens}.

Given a set of candidate solvers, our pipeline configuration algorithm can determine the optimal solver list with validate dataset. For AS-PD, the candidates are {CoT, L2M, ZeroCoT, PS, (L2M,*coarse*, (L2M,*medium*, (L2M,*fine*)} for all arithmetic datasets; {CoT, L2M, ZeroCoT, PS} for CSQA and LLC. For AS-MS, the candidates are {(GPT3.5, $n = 3$), (GPT3.5, $n = 5$), (GPT3, $n = 10$), (GPT4, n=1)}.

A.5 Statistics for Efficiency Analysis

Table 9 provides statistics on the average number of solving iterations required by various adaptive methods on the arithmetic reasoning datasets. See the analysis in 4.3.

A.6 Approach for constructing the prompt of L2M's variants in *decomposition granularity adaptation*

To illustrate, consider the following example question: *Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill?*

L2M does not control the decomposition granularity deliberately and its decomposition for the example question is as follows: *1. How much did the cappuccinos cost in total? 2. How much did the iced teas cost in total? 3. How much did the cafe lattes cost in total? 4. How much did the espressos cost in total? 5. How much did Sandy spend on drinks? 6. How much change does she receive back for a twenty-dollar bill?*

To construct L2M's variants, we first decompose

Table 8: Details of the datasets. Math: arithmetic reasoning, CS: commonsense reasoning, Sym.: symbolic reasoning.

Dataset	Domain	# Validate Samples	# Test Samples	Ave. words	Answer Type
GSM8K	Math	200	1119	46.9	Number
SVAMP	Math	200	800	31.8	Number
AQUA	Math	50	204	51.9	Option
AddSub	Math	50	345	31.5	Number
SingleEq	Math	100	408	27.4	Number
CSQA	CS	200	1021	27.8	Option
LLC	Sym.	100	400	15.0	String

Table 9: Efficiency analysis of AS-PD and AS-MS. For each method on each dataset, we count how many times of each solver in the pipeline being invoked (i.e., List of call), and then calculate the average number of solving round.

Method	Metric	GSM8K	SVAMP	CSQA	LLC	Average
AS-PD	list of #call	[1119, 338, 181, 147, 128]	[800, 236, 148, 83, 64]	[1021, 362, 114]	[400, 15]	-
	average #round	1.7	1.66	1.47	1.04	1.46
AS-MS	list of #call	[1119, 341, 205, 114]	[800, 246, 122, 69]	[1021, 385]	[400, 218, 120, 61]	-
	average #round	1.59	1.54	1.37	1.99	1.62

the question hierarchically, as shown in Figure 4.

1) First, we extract the problem and sub-problems from the first layer of decomposition. Then, serialize them from bottom to top to obtain the sequence of sub-problems in (L2M, *coarse*)'s prompt: *1. How much did Sandy spend on drinks? 2. How much change does she receive back for a twenty-dollar bill?*

2) Similarly, we extract the problem and sub-problems from the first two layers of decomposition and then serialize them to obtain the sequence of sub-problems in (L2M, *medium*)'s prompt: *1. How much did the cappuccinos cost in total? 2. How much did the iced teas cost in total? 3. How much did the cafe lattes cost in total? 4. How much did the espressos cost in total? 5. How much did Sandy spend on drinks? 6. How much change does she receive back for a twenty-dollar bill?*

3) Likewise, we extract the problem and sub-problems from the three layers of decomposition and serialize them to obtain the sequence of sub-problems in (L2M, *fine*)'s prompt: *1. How many cappuccinos did Sandy order? 2. How much did the cappuccinos cost in total? 3. How many iced teas did Sandy order? 4. How much did the iced teas cost in total? 5. How many cafe lattes did Sandy order? 6. How much did the cafe lattes cost in total? 7. How many espressos did Sandy order? 8. How much did the espressos cost in total? 9. How much did Sandy spend on all drinks in total? 10. How much change does she receive back for a twenty-dollar bill?*

A.7 Detailed Description of Automatic Pipeline Configuration

AS-MS and its pipeline configuration algorithm. The pipeline configuration of AS-MS is depicted in Figure 1(b1). The input comprises a set of solvers, wherein the variable elements include the LLM model and sample size. The output yields a pipeline of solvers aimed at minimizing costs on the validation set, while ensuring that performance decrease remains within an acceptable range. We posit that in this case solvers with higher performance tend to be more expensive. The configuration process unfolds through the following steps: 1) Designate the solver with the highest accuracy on the validation set as the last solver in the pipeline, recognizing its accuracy as the *base accuracy*. This solver is typically the most effective but also the most resource-intensive. 2) Iterate through the solvers in descending order of performance. Include a solver

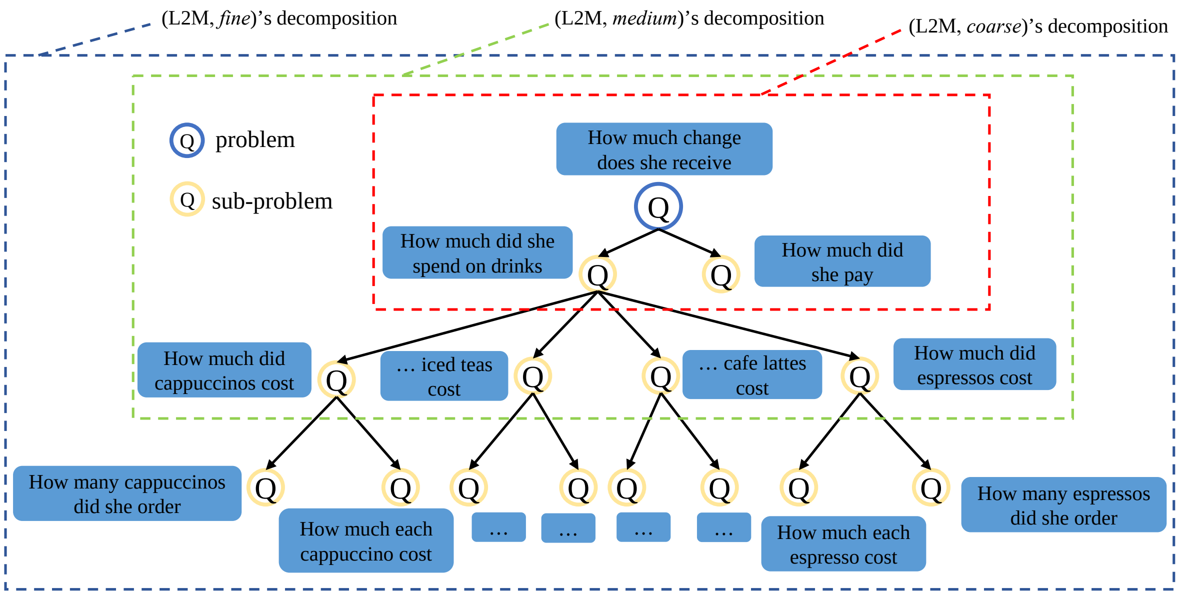


Figure 4: Illustration of hierarchical decomposition.

at the beginning of the pipeline if its inclusion does not result in an accuracy decrease beyond a specified threshold (e.g., 2%) compared to the *base accuracy*. Skip any solver that leads to exceeding this threshold, and proceed to evaluate the next one till all solvers are tried. Each solver can be used only once. Refer to Algorithm 1 for more technical details.

AS-PD and its pipeline configuration algorithm. The pipeline configuration of AS-PD is shown in Figure 1(b2). The input comprises a set of solvers, wherein the prompting method and decomposition granularity are variable elements. The output yields a pipeline of solvers aimed at maximizing overall accuracy on the validation set. The configuration process consists of the following steps: 1) Choose the solver with the highest accuracy on the validation dataset as the initial solver. 2) For the n-th (where n≥2) solver: Create a new dataset containing questions on which all the first n-1 solvers provide incorrect answers. From the remaining candidate solvers, select the one with the highest accuracy on this new dataset to append to the pipeline. If the inclusion of this solver enhances performance, incorporate it into the pipeline. Otherwise, continue testing the remaining solvers until all options have been explored. Each solver can be used only once. Refer to Algorithm 2 for more technical details.

A.8 Full sets of Prompts

We present all the prompts used in this work. For all the prompts, if we do not detect "answer is" in the response, we concatenate the question, the response and "Therefore, the answer is" to call API once gain, to obtain a short response containing the answer.

A.8.1 Zero-shot-CoT (ZeroCoT)

A.8.2 Plan-and-solve (PS)

A.8.3 Chain-of-thought (CoT) & COT_SC

A.8.4 Least-to-most (L2M)

A.8.5 L2M's variants for decomposition granularity adaptation

The following three prompts mainly differ from the decomposition granularity. For example, facing the same problem, the prompt (L2M, d_1) may break it down into 2-3 sub-questions, the prompt (L2M, d_2) may decompose it into 4-5 sub-questions, and the prompt (L2M, d_3) may decompose it into 6-8 sub-questions. In addition, the difference between them and L2M lies in: L2M lacks precise control over decomposition granularity in its demonstrations, leading to a blend of various granularities. Conversely, in the demonstrations of these variants, the decomposition granularity is either coarse, medium, or fine, depending on the specific variant.

(1) The prompts of (L2M, *coarse*)

(2) The prompts of (L2M, *medium*)

(3) The prompts of (L2M, *fine*)

Algorithm 1 Algorithm: Automatic Pipeline Configuration of AS-MS

```
Require: candidates, solver2cost_dict, reduce_thresh      ▷ We set the candidates of AS-PD as
    {("GPT3.5_SC", 3, 1), ("GPT3.5_SC", 5, 0.8), ("GPT3.5_SC", 10, 0.6), ("GPT4", 1, 1)} on all datasets
Ensure: final_acc, final_num_call_list, solver_list
  base_solver_list ← [candidates[-1]]
  base_acc, base_num_call_list ← GET_ACCURACY(base_solver_list)
  base_cost ← CALCULATE_COST(base_num_call_list, [solver2cost_dict[s] for s in base_solver_list])
  lowest_cost ← base_cost
  final_acc ← base_acc
  final_num_call_list ← base_num_call_list
  solver_list ← [candidates[-1]]
  for i in reversed(RANGE(len(candidates) - 1)) do
      solver_list ← [candidates[i]] + solver_list
      temp_acc, numCallList ← GET_ACCURACY(solver_list) ▷ numCallList records the times of being
  invoked of each solver in the pipeline
      temp_cost ← CALCULATE_COST(numCallList, [solver2cost_dict[s] for s in solver_list])
      if base_acc - temp_acc ≤ reduce_thresh and temp_cost < lowest_cost then
          lowest_cost ← temp_cost
          final_acc ← temp_acc
          final_num_call_list ← numCallList
      else
          Get rid of the first solver in solver_list
      end if
  end for
  return final_acc, final_num_call_list, solver_list
```

> **Zero-shot-CoT (ZeroCoT): Prompt for all the datasets**:
> Q: {question}
> A: Let's think step by step.

> **Plan-and-solve (PS): Prompt for all the arithmetic reasoning datasets**:
> Q: {question}
> A: Let's first understand the problem, extract relevant variables and their corresponding numerals, and make and devise a complete plan. Then, let's carry out the plan, calculate intermediate variables (pay attention to correct numerical calculation and commonsense), solve the problem step by step, and show the answer.

> **Plan-and-solve (PS): Prompt for the commonsense reasoning dataset CSQA**:
> Q: {question}
> A: Let's first prepare relevant information and make a plan. Then, let's answer the question step by step (pay attention to commonsense and logical coherence).

> **Plan-and-solve (PS): Prompt for the symbolic reasoning dataset LLC**:
> Q: {question}
> A: Let's devise a plan and solve the problem step by step.

Algorithm 2 Algorithm: Automatic Pipeline Configuration of AS-PD

Require: num_sample N, thresh θ, candidate solvers S_{cd}
Require: Solver2Records ▷ All candidate solvers's solving records on validate set
Ensure: best_solver_list

```
(Solver2crtRatio, Solver2CrtQuesSet, Solver2WrgQuesSet) ← process(S_cd, Solver2Records, N, θ
sorted_solver_crtRatio ← sort(solver2crtRatio)   ▷ Sort the candidate solvers in descending order of accuracy
solver_list ← [sorted_solver_crtRatio[0][0]]   ▷ Set the first solver as the one with highest correct ratio
left_candidates ← S_cd − solver_list[0]
WrgQuesSet ← solver2WrgQuesSet[solver_list[0]]
best_acc ← 0
best_solver_list ← solver_list
count ← len(S_cd) − 1
while count > 0 do
    count ← count − 1
    max_num ← 0
    the_solver ← None
    for solver in left_candidates do
        temp_num ← len(WrgQuesSet ∩ solver2CrtQuesSet_dict[solver])
        if temp_num > max_num then
            max_num ← temp_num
            the_solver ← solver
        end if
    end for
    if the_solver is not None then
        solver_list.append(the_solver)
        left_candidates ← left_candidates − {the_solver}
        temp_acc ← Get_accuracy(solver_list, num_sample, thresh)
        if temp_acc > best_acc then
            best_acc ← temp_acc
            best_solver_list ← solver_list
            WrgQuesSet ← WrgQuesSet − Solver2CrtQuesSet[the_solver]
        else
            solver_list.pop()
        end if
    end if
end while
return best_solver_list
```

Chain-of-thought (CoT) & COT_SC: Four-shot exemplars for all the mathematical reasoning datasets excluding the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody?
A: We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. That means that four years ago he must have been 60 - 4 = 56 years old. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32.
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy?
A: Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3.
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now?
A: Since Rena will be 5 years older than Pam in 10 years, she must be 5 years older than Pam now as well. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5.
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill?
A: Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks, which means that sandy will get $20 - $17 = $3 as change. The answer is 3.
Q: {question}
A:

Chain-of-thought (CoT) & COT_SC: Four-shot exemplars for the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody? Answer Choices: (a) 33 (b) 32 (c) 16 (d) 20
A: We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. That means that four years ago he must have been 60 - 4 = 56 years old. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32. Therefore, the answer is (b).
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy? Answer Choices: (a) 1.5 (b) 6 (c) 5 (d) 3
A: Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3. Therefore, the answer is (d).
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now? Answer Choices: (a) 5 (b) 10 (c) 15 (d) 20 (e) 30
A: Since Rena will be 5 years older than Pam in 10 years, she must be 5 years older than Pam now as well. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5. Therefore, the answer is (a).
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill? Answer Choices: (a) 20 (b) 17 (c) 3 (d) 1
A: Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks, which means that sandy will get $20 - $17 = $3 as change. The answer is 3. Therefore, the answer is (c).
Q: {question}
A:

Chain-of-thought (CoT) & COT_SC: Four-shot exemplars for the commonsense reasoning dataset CSQA:
Q: What do people use to absorb extra ink from a fountain pen? Answer Choices: (a) shirt pocket (b) calligrapher's hand (c) inkwell (d) desk drawer (e) blotter
A: The answer must be an item that can absorb ink. Of the above choices, only blotters are used to absorb ink. So the answer is (e).
Q: What home entertainment equipment requires cable? Answer Choices: (a) radio shack (b) substation (c) television (d) cabinet
A: The answer must require cable. Of the above choices, only television requires cable. So the answer is (c).
Q: The fox walked from the city into the forest, what was it looking for? Answer Choices: (a) pretty flowers (b) hen house (c) natural habitat (d) storybook
A: The answer must be something in the forest. Of the above choices, only natural habitat is in the forest. So the answer is (b).
Q: Sammy wanted to go to where the people were. Where might he go? Answer Choices: (a) populated areas (b) race track (c) desert (d) apartment (e) roadblock
A: The answer must be a place with a lot of people. Of the above choices, only populated areas have a lot of people. So the answer is (a).
Q: question
A:

Chain-of-thought (CoT) & COT_SC: Four-shot exemplars for the symbolic reasoning dataset LLC:
Q: Take the last letters of the words in "Elon Musk" and concatenate them.
A: The last letter of "Elon" is "n". The last letter of "Musk" is "k". Concatenating them is "nk". The answer is "nk".
Q: Take the last letters of the words in "Larry Page" and concatenate them.
A: The last letter of "Larry" is "y". The last letter of "Page" is "e". Concatenating them is "ye". The answer is "ye".
Q: Take the last letters of the words in "Sergey Brin" and concatenate them.
A: The last letter of "Sergey" is "y". The last letter of "Brin" is "n". Concatenating them is "yn". The answer is "yn".
Q: Take the last letters of the words in "Bill Gates" and concatenate them.
A: The last letter of "Bill" is "l". The last letter of "Gates" is "s". Concatenating them is "ls". The answer is "ls".
Q: question
A:

A.8.6 Progressive-Hint Prompting (PHP)

Least-to-most (L2M): Four-shot exemplars for all the mathematical reasoning datasets excluding the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody?
A: Let's break down this problem: 1. How old is Mohamed currently? 2. How old was Mohamed four years ago? 3. How old was Kody four years ago? 4. How old is Kody?
1. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. The answer is 60.
2. That means that four years ago he must have been 60 - 4 = 56 years old. The answer is 56.
3. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. The answer is 28.
4. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32.
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy?
A: Let's break down this problem: 1. How much did she spend on 2 bags of candy?
1. Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3.
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now?
A: Let's break down this problem: 1. How much older is Rena than Pam currently? 2. How old is Pam now?
1. Since Rena will be 5 years older than Pam in 10 years, she must be 5 years older than Pam now as well. The answer is 5.
2. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5.
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill?
A: Let's break down this problem: 1. How much did the cappuccinos cost in total? 2. How much did the iced teas cost in total? 3. How much did the cafe lattes cost in total? 4. How much did the espressos cost in total? 5. How much did Sandy spend on drinks? 6. How much change does she receive back for a twenty-dollar bill?
1. Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. The answer is 6.
2. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. The answer is 6.
3. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. The answer is 3.
4. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. The answer is 2.
5. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks. The answer is 17.
6. Sandy will get $20 - $17 = $3 as change. The answer is 3.
Q: {question}
A: Let's break down this problem:

Least-to-most (L2M): Four-shot exemplars for the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody? Answer Choices: (a) 33 (b) 32 (c) 16 (d) 20
A: Let's break down this problem: 1. How old is Mohamed currently? 2. How old was Mohamed four years ago? 3. How old was Kody four years ago? 4. How old is Kody?
1. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. The answer is 60.
2. That means that four years ago he must have been 60 - 4 = 56 years old. The answer is 56.
3. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. The answer is 28.
4. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32. Therefore, the answer is (b).
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy? Answer Choices: (a) 1.5 (b) 6 (c) 5 (d) 3
A: Let's break down this problem: 1. How much did she spend on 2 bags of candy?
1. Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3. Therefore, the answer is (d).
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now? Answer Choices: (a) 5 (b) 10 (c) 15 (d) 20 (e) 30
A: Let's break down this problem: 1. How much older is Rena than Pam currently? 2. How old is Pam now?
1. Since Rena will be 5 years older than Pam in 10 years, she must be 5 years older than Pam now as well. The answer is 5.
2. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5. Therefore, the answer is (a).
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill? Answer Choices: (a) 20 (b) 17 (c) 3 (d) 1
A: Let's break down this problem: 1. How much did the cappuccinos cost in total? 2. How much did the iced teas cost in total? 3. How much did the cafe lattes cost in total? 4. How much did the espressos cost in total? 5. How much did Sandy spend on drinks? 6. How much change does she receive back for a twenty-dollar bill?
1. Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. The answer is 6.
2. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. The answer is 6.
3. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. The answer is 3.
4. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. The answer is 2.
5. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks. The answer is 17.
6. Sandy will get $20 - $17 = $3 as change. The answer is 3. Therefore, the answer is (c).
Q: {question}
A: Let's break down this problem:

Least-to-most (L2M): Four-shot exemplars for the LLC dataset:
Q: Take the last letters of the words in "think machine" and concatenate them. A: Create sequential sublists of the list "think machine": 1. "think" 2. "think machine" Concatenate the last letters of the words within each sublist sequentially: 1. "think": The last letter of "think" is "k". 2. "think machine": "think" outputs "k". The last letter of "machine" is "e". Concatenating "k", "e" leads to "ke". The answer is "ke".
Q: Take the last letters of the words in "learning reasoning generalization" and concatenate them. A: Create sequential sublists of the list "learning reasoning generalization": 1. "learning" 2. "learning reasoning" 3. "learning reasoning generalization" Concatenate the last letters of the words within each sublist sequentially: 1. "learning": The last letter of "learning" is "g". 2. "learning reasoning": "learning" outputs "g". The last letter of "reasoning" is "g". Concatenating "g", "g" leads to "gg". 3. "learning reasoning generalization": "learning reasoning" outputs "gg". The last latter of "generalization" is "n". Concatenating "gg", "n" leads to "ggn". The answer is "ggn".
Q: Take the last letters of the words in "artificial intelligence" and concatenate them. A: Create sequential sublists of the list "artificial intelligence": 1. "artificial" 2. "artificial intelligence" Concatenate the last letters of the words within each sublist sequentially: 1. "artificial": The last letter of "artificial" is "l". 2. "artificial intelligence": "artificial" outputs "l". The last letter of "intelligence" is "e". Concatenating "l", "e" leads to "le". The answer is "le".
Q: Take the last letters of the words in "transformer language vision" and concatenate them. A: Create sequential sublists of the list "transformer language vision": 1. "transformer" 2. "transformer language" 3. "transformer language vision" Concatenate the last letters of the words within each sublist sequentially: 1. "transformer": The last letter of "transformer" is "r". 2. "transformer language": "transformer" outputs "r". The last letter of "language" is "e". Concatenating "r", "e" leads to "re". 3. "transformer language vision": "transformer language" outputs "re". The last letter of "vision" is "n". Concatenating "re", "n" leads to "ren". The answer is "ren".
Q: {question} A: Create sequential sublists of the list

(L2M, *coarse*): Four-shot exemplars for all the mathematical reasoning datasets excluding the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody?
A: Let's break down this problem: 1. How old was Kody four years ago? 2. How old is Kody?
1. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. That means that four years ago he must have been 60 - 4 = 56 years old. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. The answer is 28.
2. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32.
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy?
A: Let's break down this problem: 1. How much did she spend on 2 bags of candy?
1. Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3.
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now?
A: Let's break down this problem: 1. How much older is Rena than Pam currently? 2. How old is Pam now?
1. Since Rena will be 5 years older than Pam in 10 years, she must be 5 years older than Pam now as well. The answer is 5.
2. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5.
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill?
A: Let's break down this problem: 1. How much did Sandy spend on drinks? 2. How much change does she receive back for a twenty-dollar bill?
1. Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks. The answer is 17.
2. Sandy will get $20 - $17 = $3 as change. The answer is 3.
Q: {question}
A: Let's break down this problem:

(L2M, *coarse*): Four-shot exemplars for the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody? Answer Choices: (a) 33 (b) 32 (c) 16 (d) 20
A: Let's break down this problem: 1. How old was Kody four years ago? 2. How old is Kody?
1. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. That means that four years ago he must have been 60 - 4 = 56 years old. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. The answer is 28.
2. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32. Therefore, the answer is (b).
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy? Answer Choices: (a) 1.5 (b) 6 (c) 5 (d) 3
A: Let's break down this problem: 1. How much did she spend on 2 bags of candy?
1. Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3. Therefore, the answer is (d).
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now? Answer Choices: (a) 5 (b) 10 (c) 15 (d) 20 (e) 30 A: Let's break down this problem: 1. How much older is Rena than Pam currently? 2. How old is Pam now? 1. Since Rena will be 5 years older than Pam in 10 years, she must be 5 years older than Pam now as well. The answer is 5.
2. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5. Therefore, the answer is (a).
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill? Answer Choices: (a) 20 (b) 17 (c) 3 (d) 1
A: Let's break down this problem: 1. How much did Sandy spend on drinks? 2. How much change does she receive back for a twenty-dollar bill?
1. Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks. The answer is 17.
2. Sandy will get $20 - $17 = $3 as change. The answer is 3. Therefore, the answer is (c).
Q: question
A: Let's break down this problem:

(L2M, *medium*): Four-shot exemplars for all the mathematical reasoning datasets excluding the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody?
A: Let's break down this problem: 1. How old was Mohamed four years ago? 2. How old was Kody four years ago? 3. How old is Kody?
1. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. That means that four years ago he must have been 60 - 4 = 56 years old. The answer is 56.
2. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. The answer is 28.
3. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32.
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy?
A: Let's break down this problem: 1. How much did she spend on 1 bag of candy? 2. How much did she spend on 2 bags of candy?
1. Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50.
2. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3.
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now?
A: Let's break down this problem: 1. How much older is Rena than Pam in 10 years? 2. How much older is Rena than Pam currently? 3. How old is Pam now?
1. We are told that Rena will be 5 years older than Pam in 10 years. The answer is 5.
2. So she must be 5 years older than Pam now as well. The answer is 5.
3. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5.
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill?
A: Let's break down this problem: 1. How much did the cappuccinos cost in total? 2. How much did the iced teas cost in total? 3. How much did the cafe lattes cost in total? 4. How much did the espressos cost in total? 5. How much did Sandy spend on drinks? 6. How much change does she receive back for a twenty-dollar bill?
1. Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. The answer is 6.
2. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. The answer is 6.
3. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. The answer is 3.
4. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. The answer is 2.
5. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks. The answer is 17.
6. Sandy will get $20 - $17 = $3 as change. The answer is 3.
Q: {question}
A: Let's break down this problem:

(L2M, *medium*): Four-shot exemplars for the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody? Answer Choices: (a) 33 (b) 32 (c) 16 (d) 20
A: Let's break down this problem: 1. How old was Mohamed four years ago? 2. How old was Kody four years ago? 3. How old is Kody?
1. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. That means that four years ago he must have been 60 - 4 = 56 years old. The answer is 56.
2. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. The answer is 28.
3. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32. Therefore, the answer is (b).
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy? Answer Choices: (a) 1.5 (b) 6 (c) 5 (d) 3
A: Let's break down this problem: 1. How much did she spend on 1 bag of candy? 2. How much did she spend on 2 bags of candy?
1. Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50.
2. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3. Therefore, the answer is (d).
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now? Answer Choices: (a) 5 (b) 10 (c) 15 (d) 20 (e) 30
A: Let's break down this problem: 1. How much older is Rena than Pam in 10 years? 2. How much older is Rena than Pam currently? 3. How old is Pam now?
1. We are told that Rena will be 5 years older than Pam in 10 years. The answer is 5.
2. So she must be 5 years older than Pam now as well. The answer is 5.
3. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5. Therefore, the answer is (a).
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill? Answer Choices: (a) 20 (b) 17 (c) 3 (d) 1
A: Let's break down this problem: 1. How much did the cappuccinos cost in total? 2. How much did the iced teas cost in total? 3. How much did the cafe lattes cost in total? 4. How much did the espressos cost in total? 5. How much did Sandy spend on drinks? 6. How much change does she receive back for a twenty-dollar bill?
1. Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. The answer is 6.
2. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. The answer is 6.
3. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. The answer is 3.
4. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. The answer is 2.
5. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks. The answer is 17.
6. Sandy will get $20 - $17 = $3 as change. The answer is 3. Therefore, the answer is (c).
Q: question
A: Let's break down this problem:

(L2M, *fine*): Four-shot exemplars for all the mathematical reasoning datasets excluding the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody?
A: Let's break down this problem: 1. How old is Mohamed currently? 2. How old was Mohamed four years ago? 3. How old was Kody four years ago? 4. How old is Kody?
1. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. The answer is 60.
2. That means that four years ago he must have been 60 - 4 = 56 years old. The answer is 56.
3. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. The answer is 28.
4. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32.
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy?
A: Let's break down this problem: 1. What was the original price of each bag? 2. What was the discount ratio of each bag? 3. How much did she spend on 1 bag of candy? 4. How much did she spend on 2 bags of candy?
1. Each bag was originally $6.00. The answer is 6.
2. But it was 75% off. The answer is 0.75.
3. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. The answer is 1.50.
4. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3.
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now?
A: Let's break down this problem: 1. How much older is Rena than Pam in 10 years? 2. How much older is Rena than Pam currently? 3. Which equations need to be constructed to calculate Pam's age? 4. How old is Pam now?
1. We are told that Rena will be 5 years older than Pam in 10 years. The answer is 5.
2. So she must be 5 years older than Pam now as well. The answer is 5.
3. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. The answer is R = 2 * P and R = P + 5.
4. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5.
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill?
A: Let's break down this problem: 1. How many cappuccinos did Sandy order? 2. How much did the cappuccinos cost in total? 3. How many iced teas did Sandy order? 4. How much did the iced teas cost in total? 5. How many cafe lattes did Sandy order? 6. How much did the cafe lattes cost in total? 7. How many espressos did Sandy order? 8. How much did the espressos cost in total? 9. How much did Sandy spend on all drinks in total? 10. How much change does she receive back for a twenty-dollar bill?
1. Sandy ordered three cappuccinos. The answer is 3.
2. Each cappuccino cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. The answer is 6.
3. She ordered two iced teas. The answer is 2.
4. Each iced tea cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. The answer is 6.
5. She ordered two cafe lattes. The answer is 2.
6. Each cafe latte cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. The answer is 3.
7. She ordered two espressos. The answer is 2.
8. Each espressos cost $1 each, so she spent $1 * 2 = $2 on espressos. The answer is 2.
9. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks. The answer is 17.
10. Sandy will get $20 - $17 = $3 as change. The answer is 3.
Q: {question}
A: Let's break down this problem:

(L2M, *fine*): Four-shot exemplars for the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody? Answer Choices: (a) 33 (b) 32 (c) 16 (d) 20
A: Let's break down this problem: 1. How old is Mohamed currently? 2. How old was Mohamed four years ago? 3. How old was Kody four years ago? 4. How old is Kody?
1. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. The answer is 60.
2. That means that four years ago he must have been 60 - 4 = 56 years old. The answer is 56.
3. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. The answer is 28.
4. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32. Therefore, the answer is (b).
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy? Answer Choices: (a) 1.5 (b) 6 (c) 5 (d) 3
A: Let's break down this problem: 1. What was the original price of each bag? 2. What was the discount ratio of each bag? 3. How much did she spend on 1 bag of candy? 4. How much did she spend on 2 bags of candy?
1. Each bag was originally $6.00. The answer is 6.
2. But it was 75% off. The answer is 0.75.
3. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. The answer is 1.50.
4. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3. Therefore, the answer is (d).
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now? Answer Choices: (a) 5 (b) 10 (c) 15 (d) 20 (e) 30
A: Let's break down this problem: 1. How much older is Rena than Pam in 10 years? 2. How much older is Rena than Pam currently? 3. Which equations need to be constructed to calculate Pam's age? 4. How old is Pam now?
1. We are told that Rena will be 5 years older than Pam in 10 years. The answer is 5.
2. So she must be 5 years older than Pam now as well. The answer is 5.
3. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. The answer is R = 2 * P and R = P + 5.
4. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5. Therefore, the answer is (a).
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill? Answer Choices: (a) 20 (b) 17 (c) 3 (d) 1
A: Let's break down this problem: 1. How many cappuccinos did Sandy order? 2. How much did the cappuccinos cost in total? 3. How many iced teas did Sandy order? 4. How much did the iced teas cost in total? 5. How many cafe lattes did Sandy order? 6. How much did the cafe lattes cost in total? 7. How many espressos did Sandy order? 8. How much did the espressos cost in total? 9. How much did Sandy spend on all drinks in total? 10. How much change does she receive back for a twenty-dollar bill?
1. Sandy ordered three cappuccinos. The answer is 3.
2. Each cappuccino cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. The answer is 6.
3. She ordered two iced teas. The answer is 2.
4. Each iced tea cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. The answer is 6.
5. She ordered two cafe lattes. The answer is 2.
6. Each cafe latte cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. The answer is 3.
7. She ordered two espressos. The answer is 2.
8. Each espressos cost $1 each, so she spent $1 * 2 = $2 on espressos. The answer is 2.
9. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks. The answer is 17.
10. Sandy will get $20 - $17 = $3 as change. The answer is 3. Therefore, the answer is (c).
Q: question
A: Let's break down this problem:

Progressive-Hint Prompting (PHP): Four-shot exemplars for the mathematical reasoning datasets excluding the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody? (Hint: The answer is near to 32).
A: We know the Answer Hints: 32. With the Answer Hints: 32, we will answer the question. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. That means that four years ago he must have been 60 - 4 = 56 years old. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is 32.
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy? (Hint: The answer is near to 2, 5).
A: We know the Answer Hints: 2, 5. With the Answer Hints: 2, 5, we will answer the question. Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is 3.
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now? (Hint: The answer is near to 10, 5).
A: We know the Answer Hints: 10, 5. With the Answer Hints: 10, 5, we will answer the question. Since Rena will be 5 years older than Pam in 10 years, she must be 5 years older than Pam now as well. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is 5.
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill? (Hint: The answer is near to 3).
A: We know the Answer Hints: 3. With the Answer Hints: 3, we will answer the question. Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks, which means that sandy will get $20 - $17 = $3 as change. The answer is 3.

Progressive-Hint Prompting (PHP): Four-shot exemplars for the AQuA dataset:
Q: Four years ago, Kody was only half as old as Mohamed. If Mohamed is currently twice 30 years old, how old is Kody? (Hint: The answer is near to (b)). Answer Choices: (a) 33 (b) 32 (c) 16 (d) 20
A: We know the Answer Hints: (b). With the Answer Hints: (b), we will answer the question. We were told that Mohamed is currently twice 30 years old, so he is currently 30 * 2 = 60 years old. That means that four years ago he must have been 60 - 4 = 56 years old. Four years ago, Kody was half as old as Mohamed, so Kody must have been 56 / 2 = 28 years old then. Since Kody was 28 years old four years ago, she must now be 28 + 4 = 32 years old. The answer is (b).
Q: Carla bought 2 bags of mini peanut butter cups on clearance. Each bag was $6.00 but was 75% off. How much did she spend on 2 bags of candy? (Hint: The answer is near to (a), (c)). Answer Choices: (a) 1.5 (b) 6 (c) 5 (d) 3
A: We know the Answer Hints: (a), (c). With the Answer Hints: (a), (c), we will answer the question. Each bag was $6.00 but was 75% off. So each bag cost $6.00 * (1 - 0.75) = $6.00 * 0.25 = $1.50. Carla bought 2 bags. So she spent $1.50 * 2 = $3.00. The answer is (d).
Q: If Pam is currently twice as young as Rena is, and in 10 years Rena will be 5 years older than her, how old is Pam now? (Hint: The answer is near to (a), (b)). Answer Choices: (a) 5 (b) 10 (c) 15 (d) 20 (e) 30
A: We know the Answer Hints: (a), (b). With the Answer Hints: (a), (b), we will answer the question. Since Rena will be 5 years older than Pam in 10 years, she must be 5 years older than Pam now as well. If Pam is currently twice as young as Rena, that means that Rena is currently twice as old as Pam is. So if P stands for Pam's age now and R stands for Rena's age now, then we know that R = 2 * P And since Rena is 5 years older than Pam now, we know that R = P + 5. By substitution, we have P + 5 = 2 * P, which means that P = 5. The answer is (a).
Q: Cappuccinos cost $2, iced teas cost $3, cafe lattes cost $1.5 and espressos cost $1 each. Sandy orders some drinks for herself and some friends. She orders three cappuccinos, two iced teas, two cafe lattes, and two espressos. How much change does she receive back for a twenty-dollar bill? (Hint: The answer is near to (c)). Answer Choices: (a) 20 (b) 17 (c) 3 (d) 1
A: We know the Answer Hints: (c). With the Answer Hints: (c), we will answer the question. Sandy ordered three cappuccinos, which cost $2 each, so she spent $2 * 3 = $6 on cappuccinos. She ordered two iced teas, which cost $3 each, so she spent $3 * 2 = $6 dollars on ice teas. She ordered two cafe lattes, which cost $1.5 each, so she spent $1.5 * 2 = $3 on cafe lattes. She ordered two espressos, which cost $1 each, so she spent $1 * 2 = $2 on espressos. So altogether, Sandy spent $6 + $6 + $3 + $2 = $17 on drinks, which means that sandy will get $20 - $17 = $3 as change. The answer is (c).