Assimilation and Accommodation: Task-Adaptive Hierarchical Abstraction for Solving Web Tasks

Xinyu Pang[1]*, **Ruixin Hong**[1,2]*, **Hongming Zhang**[2], **Changshui Zhang**[1]†
[1]Institute for Artificial Intelligence, Tsinghua University (THUAI);
[1]Beijing National Research Center for Information Science and Technology (BNRist);
[1]Department of Automation, Tsinghua University, Beijing, P.R.China
[2]Tencent AI Lab, Seattle
{pangxy22, hrx20}@mails.tsinghua.edu.cn,
hongmzhang@global.tencent.com,
zcs@mail.tsinghua.edu.cn

Abstract

Web tasks, which involve processing data from online resources, challenge agents to generalize beyond fixed knowledge to unseen task contexts. Learning from experience, the ability to derive reusable patterns from past tasks, is crucial for improving generalization. However, existing methods focus on summarizing workflows, *i.e.*, common sub-routines, which may introduce excessive low-level details that distract models. Additionally, the absence of task-specific objectives can lead to inconsistencies between workflows and future task queries, hindering reasoning performance. This paper seeks to mitigate these issues by proposing A^2, a framework that derives task-adaptive hierarchical abstraction to enhance web task reasoning. Our approach first extracts general-purpose semantic abstraction from past task-solution pairs. Combined with the next task query, this abstraction forms a task-adaptive episodic abstraction that guides subsequent reasoning. Experiments show that A^2 achieves superior performance with competitive cost-efficiency, improving success rates by 0.7% on Mind2web and 4.6% on Webarena. The code is accessible at https://github.com/Xinyu-Pang/A2.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language reasoning tasks (Brown et al., 2020; OpenAI, 2023; Touvron et al., 2023). Despite the promising performance, state-of-the-art LLMs still struggle with web tasks (Yao et al., 2022a), which involve processing dynamic online information. The primary difficulty arises from the evolving nature of web content. Such variability often exceeds the static knowledge stored in pre-trained models, posing significant challenges for generalization.

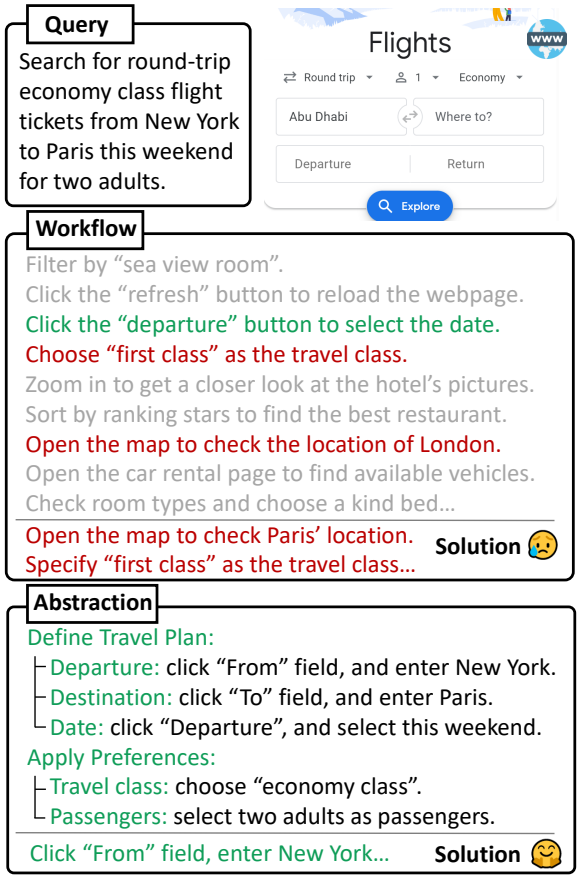

Figure 1: Comparison of workflow and abstraction. Workflow may include irrelevant (gray) or misleading (red) details, causing incorrect solutions, whereas task-adaptive hierarchical abstraction provides focused and effective guidance (green), ensuring accurate solutions.

One approach to addressing these challenges is *learning from experience*, i.e., identifying reusable patterns shared across similar tasks and generalizing them to novel environments. Prior efforts have explored to learn from experience by summarizing workflows (Wang et al., 2024), as illustrated in Fig. 1. These works derive common sub-routines from experiential data, capturing the execution logic of each step to assist further reasoning.

However, summarizing workflow is not universally effective, primarily due to two issues. (1)

*Equal contribution.
†Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2025, pages 14000–14014

As experiential data grows, workflow memory expands, escalating computational costs and potentially exceeding context window limits (Liu et al., 2024b; Zhao et al., 2024). The accumulation of irrelevant details distracts the model, hindering key insight extraction. Consequently, reasoning performance deteriorates as the model becomes overwhelmed by nonessential information. For instance, irrelevant workflows (gray) in Fig. 1 may obscure critical ones (green), leading to incorrect solutions. (2) The absence of a specific target task during workflow summarization may yield content misaligned with subsequent tasks, leading to suboptimal or even erroneous outcomes. As shown in Fig. 1, misleading workflows (red) may make models incorrectly prioritize first class over economy.

A key distinction between LLMs and human reasoning lies in the ability to form *task-adaptive hierarchical abstraction* from experience. As demonstrated in Fig. 1, it involves organizing experiential data into structured layers, where higher-level concepts build upon lower-level details. When faced with novel environments, general-purpose abstraction transitions into task-adaptive abstraction, enabling context-specific guidance. Humans naturally develop this ability early, enabling them to adapt prior knowledge to solve novel challenges (Mitchelmore, 2002; Fine, 2002). Compared to workflow-based approaches, it provides a more structured approach to distill key patterns and guide effective reasoning. Despite its potential, task-adaptive hierarchical abstraction remains unexplored in LLMs.

In this paper, we propose to enhance LLMs' web task reasoning capability by developing task-adaptive hierarchical abstraction from experience (Sec. 4). We introduce **A**ssimilation and **A**ccommodation (A^2), consisting of three key steps. (1) Assimilation: General-purpose semantic abstraction is derived from previously solved tasks, leveraging knowledge to guide reasoning. (2) Accommodation: This abstraction is refined into a task-specific episodic one based on the new task query, ensuring alignment with task requirements. (3) Utilization: The episodic abstraction then guides reasoning to improve performance.

Extensive experiments show that A^2 achieves superior performance with remarkable cost-efficiency. On Webarena (Zhou et al., 2024), a web task benchmark, it improves success rate by 4.6% *w.r.t.* baselines using GPT-4o-mini. On Mind2web (Deng et al., 2023), a web navigation dataset, it outperforms state-of-the-art methods with a 4.5% gain in step success rate and a 0.7% increase in overall success rate. Further analysis confirms our method's robustness, indispensability, and lower token cost, highlighting its potential for web task resolution.

2 Related Work

Web tasks, as the research focus of this work, refer to reasoning tasks that involve web data, such as retrieving or processing information from online sources. Current representative methods to addressing web tasks tend to prompt LLMs (Zheng et al., 2024; Zhou et al., 2024), or train agents by reinforcement learning (Humphreys et al., 2022; Liu et al., 2018). Various web task benchmarks have been introduced to evaluate the performance of these approaches (Liu et al., 2018; Yao et al., 2022a; Zhou et al., 2024). Web tasks pose unique challenges compared to other reasoning problems, characterized by dynamic data dependencies and the need for multi-step action sequences. These requirements demand both generalization from prior knowledge and continuous adaptation. In this work, we address these challenges through a task-adaptive hierarchical abstraction framework, offering a robust and scalable solution for web-based reasoning.

Inductive reasoning seeks to derive common principles from specific instances. Typically, LLMs are provided with 2–5 input-output pairs that adhere to a pre-defined rule and are then expected to deduce the underlying rule and solve a corresponding test problem. Existing research mainly focus on prompt-based strategies (Yang et al., 2024b; Mirchandani et al., 2023; Xu et al., 2023), with dedicated benchmarks assessing these capabilities (Chollet, 2019; Moskvichev et al., 2023). However, such works are limited by the reliance on artificial, context-specific rules derived from minimal examples, hindering real-world applicability. Our approach addresses this by leveraging task-adaptive hierarchical abstraction to enhance generalization, enabling LLMs to tackle practical challenges beyond constrained example-based scenarios.

Utilization of experience. To boost reasoning performance, models are expected to make full use of known information, *i.e.*, previous tasks and model-generated solutions. In general, current approaches can fall into three main categories: (1) *Few-shot ICL* uses previous queries and solutions as demonstrations in the prompts (Dong et al., 2024; Song et al., 2023). However, while intuitively appealing, it suffers from implicit knowledge transfer and context window limitations. (2) *Rule summa-*

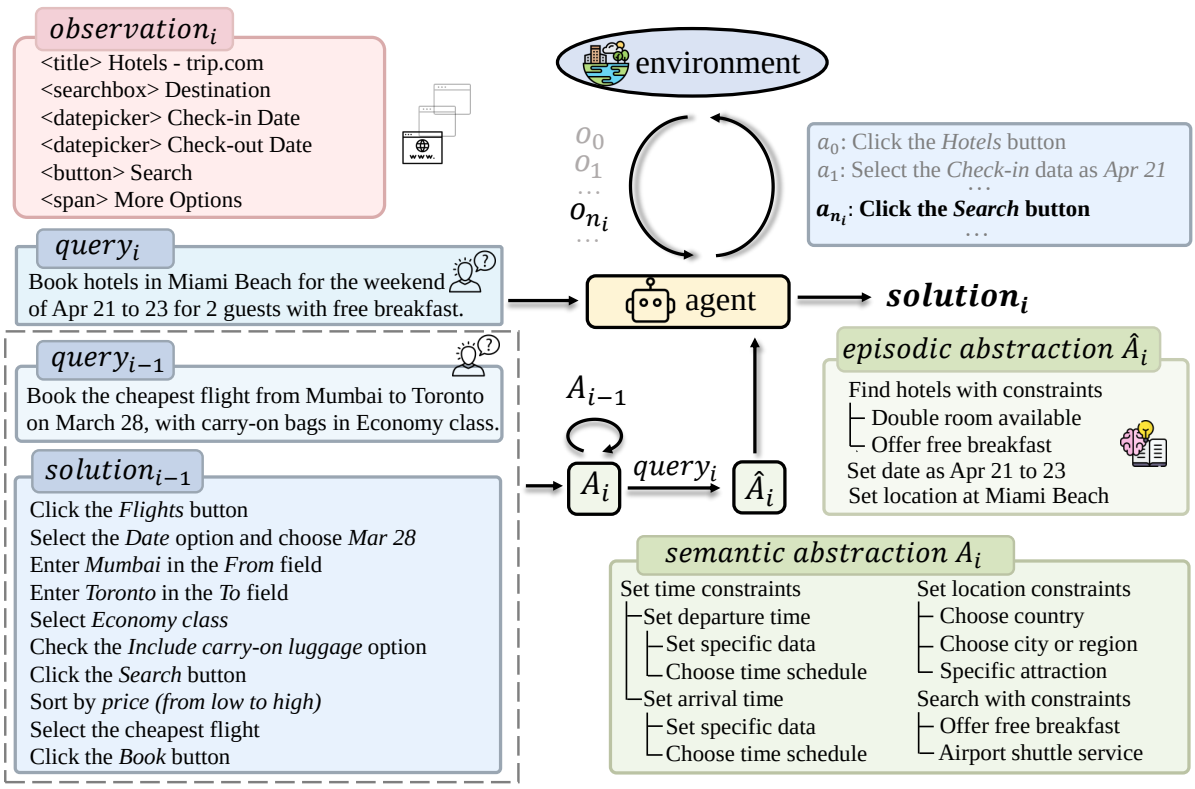

Figure 2: Illustration of A^2, solving web tasks using LLMs with the help of task-adaptive hierarchical abstraction from experience. The approach consists of three key stages: assimilation, accommodation, and utilization.

rization extracts general rules from past tasks (Zhu et al., 2023). However, these rules typically lack detailed execution instructions and fail to adapt effectively to novel task contexts. (3) *Workflow or template summarization* derives common execution patterns, predominantly through prompting (Wang et al., 2024; Zheng et al., 2024; Yang et al., 2024a), or rule-based techniques (Ellis et al., 2020; Bowers et al., 2023). However, two critical limitations persist: (1) The lack of specific summarization targets often results in irrelevant or even misleading knowledge for subsequent tasks. (2) As past tasks expand, the accumulation of extraneous details may distract models from effectively extracting key insights necessary for guiding reasoning processes.

3 Task Definition

In this section, we build upon representative works (Yao et al., 2022a,b) to formally define web tasks as tasks that necessitate interaction with web-based environments, including retrieving, processing, and reasoning over data from online resources. Agents generate action sequences as solutions based on task queries and the dynamic states of websites. The inclusion of numerous and constantly evolving HTML elements presents significant challenges for existing methods. Web task benchmarks generally follow a sequential task order, requiring methods to adopt an online approach, solving tasks one by one. Due to the significant differences between websites, tasks are typically grouped by website, with all tasks for a given website completed before proceeding to the next one.

As shown in Fig. 3, given a task query q and an initial state $S = \emptyset$, the agent first gathers an initial observation o_0 from the environment E. Using the observation o_0, the current state S, and the task query q, the agent predicts an action $a_0 = f(q, S, o_0)$. After executing a_0, the state is updated to $S = S \cup \{(o_0, a_0)\}$, and a new observation $o_1 = h(E, S)$ is collected from the environment, where h represents the environment update function. This process repeats until q is satisfactorily resolved or a pre-defined maximum step limit m is attained. The final solution is represented as a sequence of executed actions, denoted by $s = \{(a_0, a_i, \ldots, a_n)\}$. For a given website, the set of tasks $Q = \{q_0, q_1, \ldots, q_K\}$ is solved sequentially, ensuring that all tasks for one website are completed before proceeding to the next website.

4 Method

In this section, we introduce A^2, a framework designed to enhance LLMs' capability to learn from

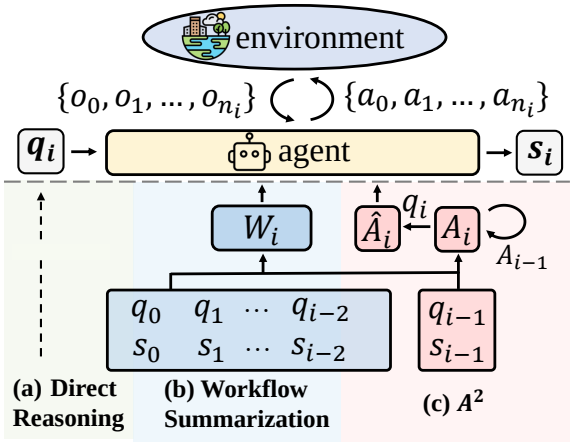

Figure 3: Comparison of three reasoning methods: (a) direct reasoning without prior knowledge, (b) workflow summarization, and (c) our reasoning framework A^2.

experience and adapt to novel environments.

As illustrated in Fig. 2, A^2 is built on the design principles of discovery and utilization, i.e., (i) ***assimilation***: dynamically extracting general-purpose hierarchical semantic abstraction from experiential data (Sec. 4.2); (ii) ***accommodation***: adapting semantic abstraction to new queries, generating task-adaptive episodic abstraction as reasoning guidance (Sec. 4.3); and (iii) ***utilization***: applying episodic abstraction to steer problem-solving process (Sec. 4.4). Details are provided below.

Existing approaches generally follow a sequential procedure for each website (Deng et al., 2023; Wang et al., 2024), as shown in Fig. 3(b). Given a set of previously solved query-solution pairs $P = \{(q_0, s_0), (q_1, s_1), \ldots, (q_{i-1}, q_{i-1})\}$, workflows $W = \{w_0, w_1, ..., w_n\}$ are derived. Each workflow $w_j (0 \leq j \leq n)$ represents a set of common sub-routines extracted from historical data. For a new task query q_i, the model generates a solution $s_i = \{a_0, a_1, ..., a_{n_i}\}$ guided by W, where each a_j represents an action (detailed in Sec. 3). The new query-solution pair (q_i, s_i) is added to update solved tasks $P' = P \cup \{(q_i, s_i)\}$, and workflows are updated. This process repeats until all tasks for the current website are completed.

4.1 Overview

Our approach consists of three stages, as shown in Fig. 3 (c). (1) To leverage experience from previously solved tasks, we derive a general-purpose *semantic abstraction* from past tasks. Prior practices that involve all solved tasks may introduce irrelevant or misleading details, as discussed in Sec. 1. Instead, we incrementally update the semantic abstraction using the most recent query-solution pair:

Algorithm 1 A^2: Task-Adaptive Abstraction

Require: Reasoning agent f, abstraction agent g, task set $Q = \{q_0, q_1, \ldots, q_K\}$, maximum steps m, environment E, environment update function h

Initialize semantic abstraction $A_0 \leftarrow \emptyset$, episodic abstraction $\tilde{A}_0 \leftarrow \emptyset$
for $i = 0 : K$ **do**
  Initialize state $S \leftarrow \emptyset$
  if $i > 0$ **then** ▷ Step-1: Assimilation
    $A_i \leftarrow g(A_{i-1}, q_{i-1}, s_{i-1})$
  end if
  $\hat{A}_i \leftarrow g(A_i, q_i)$ ▷ Step-2: Accommodation
  for $k = 0 : m$ **do** ▷ Step-3: Utilization
    Gather observation $o_k \leftarrow h(E, S)$
    Predict action $a_k \leftarrow f(q_i, o_k, S)$
    if $a_k =$ None **then**
      Break
    end if
    Update state $S' \leftarrow S \cup \{(o_k, a_k)\}$
  end for
  yield solution $s_i \leftarrow \{a_0, a_1, \ldots, a_{n_i}\}$
end for

$A_i = g(A_{i-1}, q_{i-1}.s_{i-1})$ where A represents semantic abstraction, g represents abstraction agent. (2) We adapt it to the next task query q_i, forming a task-specific *episodic abstraction*: $\hat{A}_i = g(A_i, q_i)$, ensuring relevance between abstraction and the current task. The process mirrors human cognitive mechanisms, where episodic abstraction encodes task-specific and contextualized knowledge. Both A_i and $\hat{A}_i$ are hierarchically structured, capturing knowledge at varying levels of granularity. A_i provides general-purpose insights, while $\hat{A}_i$ refines it for the current task. (3) Then $\hat{A}_i$ guides the reasoning process to solve q_i. The process iterates for each new task, dynamically updating the semantic abstraction and generating a task-specific episodic abstraction. The pipeline is shown in Algorithm. 1.

4.2 Assimilation

Web tasks involve diverse task contexts and dynamic data, which frequently surpass models' pre-trained knowledge, posing challenges in effective reasoning. Given limited information, *i.e.*, previously solved queries and model-generated solutions, a practical way is learning general knowledge from experience, akin to human cognitive process. According to Jean Piaget's cognitive development theory (Piaget, 1977, 2013), we define the stage of

forming semantic abstraction with general-purpose knowledge as **assimilation**, which integrates novel information into existing schema. In our framework, assimilation constructs semantic abstraction A that stores general-purpose abstraction knowledge derived from previously solved tasks.

Technically, during the assimilation stage, we expect LLMs to extract semantic abstraction with general purposes, representing widely-purposed knowledge from previous experiential data. After solving the $(i-1)^{\text{th}}$ task, its query and model-generated solution are provided to supplement and adjust existing semantic abstraction A_{i-1}, producing A_i in an on-policy process. To produce an abstraction that more accurately captures the implicit patterns or principles across previously completed tasks, we propose an LM-based module that prompts the LLM to extract and derive hierarchical abstraction from the prior input experiences, *i.e.*, previously solved tasks and corresponding predicted solutions.

Abstraction Structure. Each piece of abstraction consists of two key components: (1) a natural language goal specification and (2) executable implementation procedures, including the operation type, the content of involved elements, and element IDs. For instance, the "modify price" abstraction combines the description "Adjust product pricing" with concrete operations like "fill([element id], [new price])" and "click save button [element id]." The textual description explicitly defines the abstraction's purpose, while the procedural steps provide precise guidance for the execution process.

Like human memory structure (Anderson, 2013), A_i is organized in a hierarchical manner, therein higher levels encapsulate the functionality and objectiveness of lower levels. For example, in tasks such as booking hotels or flights, low-level actions like searching, selecting, and confirming can be grouped into higher-level abstractions such as planning, booking, and confirmation. The depth is dynamically adjusted according to the comprehensiveness of the experiential data and the desired granularity for each task. This hierarchical organization captures multi-grained patterns, balancing fine-grained details with overarching logic to enhance knowledge retrieval and application while minimizing distraction from excessive information.

4.3 Accommodation

Although A_i embodies extensive general-purpose world knowledge accumulated from a wide range of tasks, it may not precisely align with the specific demands of a new query q_i. This misalignment arises because A_i often includes extraneous information irrelevant to the current task. The primary challenge in facilitating effective reasoning lies in efficiently utilizing the knowledge within the ongoing reasoning process. We would claim that deriving query-adaptive knowledge is of the essence. Inspired by cognitive theory (Piaget, 1977, 2013), we term the process **accommodation**, wherein existing knowledge structures are modified or reconstructed to integrate and adapt to novel information – specifically, the next query q_i in this context.

To achieve this, we instantiate semantic abstraction A_i along with the next query q_i into episodic abstraction $\hat{A}_i$, which is adapted from A_i but retains only query-relevant knowledge. $\hat{A}_i$ is highly task-adaptive and well-suited for guiding subsequent query reasoning. The adaptation ensures that the abstraction remains focused and pertinent, thus mitigating potential misguidance caused by indiscriminate summarization. Assimilation integrates new information into existing schema, while accommodation entails modifying existing schema to adapt to novel information. The dynamic balance between these processes enables both precise execution and broad generalization. In our framework, assimilation forms semantic abstraction that stores general-purpose world knowledge, whereas accommodation adapts it into episodic abstraction tailored to specific queries. The semantic abstraction is stored and updated in the reasoning process, while the episodic abstraction serves as a pull-and-play assistance, providing targeted assistance.

4.4 Utilization

Following the assimilation and accommodation stages, the task-adaptive episodic abstraction $\hat{A}_i$ is constructed, functioning as a succinct yet informative representation to steer the ensuring reasoning processes. Since the derived episodic abstraction is compact and focused on only the most relevant abstraction, the entirety of $\hat{A}_i$ is incorporated into the subsequent reasoning procedure. Specifically, we integrate three components for reasoning: episodic abstraction, task query, and observation from the website environment. Then the agents are required to predict the next action to execute. Further details on the utilization process are provided in Sec. 3.

5 Experiments

This section presents a comprehensive evaluation of A^2 with representative baselines on several web

task datasets. Firstly, we introduce several critical parts of the experimental setups (in Sec. 5.1). Secondly, we provide a comprehensive analysis (in Sec. 5.2). Lastly, we conduct extensive analysis studies to deeply understand the task-adaptive hierarchical abstraction (in Sec. 5.3 to Sec. 5.5).

5.1 Implementation

• **Datasets.** We evaluate A^2 with baselines on two web benchmarks: Mind2web and Webarena. *Mind2web* (Deng et al., 2023) serves as a benchmark assessing the ability to follow instructions for completing tasks on real-world websites. Statistics are shown in Table 5 in the appendix. It consists of tasks from 137 websites across 31 domains. These tasks are systematically divided into three subsets based on their relationship with the training set: (1) cross-task (tasks from encountered websites, 252 instances), (2) cross-website (tasks from unseen websites, 177 instances), and (3) cross-domain (tasks from entirely novel domains, 912 instances). *Webarena* (Zhou et al., 2024) operates as a self-hostable web environment designed for developing autonomous agents and creating websites with real-world functionality and data. To simulate human problem-solving, it integrates tools and knowledge resources as standalone websites. The dataset features 812 long-horizon tasks, grouped into five classes according to website types. Both datasets are highly challenging and rich in real-world scenarios. Solving these tasks requires models to effectively leverage prior knowledge, adapt to diverse contexts, and engage in dynamic reasoning.

• **Metrics.** To ensure a comprehensive evaluation, we consider distinct metrics for these datasets. In the Mind2web dataset, evaluation metrics consider both elements and whole tasks, comprising four metrics in total. (1) *Element Accuracy (EA)* measures the match between the selected element and all acceptable options. (2) *Operation F1 (AF)* calculates the token-level F1 score for the predicted operation. For the whole task, (3) *Step Success Rate (Step SR)* and (4) *Success Rate (SR, for the whole task)* are used. A step is deemed successful if both the predicted element and operation are correct. A task is successful if all steps succeed. *Success Rate (SR)* is the metric in Webarena.

• **Baselines.** We select multiple representative baseline methods to ensure a comprehensive evaluation, following the A^2 evaluation paradigm to ensure a rigorous evaluation process:

• **MindAct (Deng et al., 2023)** tackles the Mind2web benchmark by scoring promising elements using small language models and retaining high-potential promising elements for subsequent reasoning. It processes tasks directly with instructions, without prior knowledge or demonstrations. It is exclusively applied to Mind2web.

• **Trajectory-as-Exemplar (TaE)** (Zheng et al., 2024) employs human-designed query-solution pairs as few-shot demonstrations, with solutions as action trajectories. As MindAct is specific to Mind2web, TaE serves as the Webarena baseline for fair "no prior knowledge" comparison.

• **Agent Workflow Memory (AWM)** (Wang et al., 2024), a method for inducing commonly reused sub-routines to guide subsequent generations.

• **Implementation Details.** Due to time and cost constraints, we primarily employ GPT-4o-mini (OpenAI, 2024) as the base LLM for the analyses in this study. We also include deepseek-v3 (Liu et al., 2024a) in our experiments for additional comparison. Due to the significant variation in website environments and the tasks in web task benchmarks are generally sequential, we follow standard practice by independently summarizing abstraction knowledge for each website. Abstraction knowledge is not shared across different websites to prevent possible misleading effects. For the Mind2web dataset, we follow Deng et al. (2023) and use a fine-tuned small model to score raw HTML elements, narrowing down the pool of potential candidates. Specifically, we use a trained DeBERTa-v3-base model with 86M parameters to rank elements, selecting the top five for subsequent reasoning. Browsergym (Drouin et al., 2024) is adopted for Webarena for efficient evaluation.

For all baselines and our approach, we consistently set the **temperature** parameter to zero to minimize output variability and ensure reproducibility. For evaluations on Webarena, we adopt the configuration used in AWM and set the temperature to 1.0. The **maximum token** limit and **stop tokens** adhere to their default settings. Given the characteristics of the datasets, we apply different **task selection criteria** for evaluating success. For Webarena, the semantic abstraction is updated when the previous task is successfully solved, ensuring the accuracy of the abstraction. In contrast, for Mind2web, due to the limited number of tasks per website, we do not require the previously solved task to be solved correctly before updating the abstraction. These approaches ensure the accumulation of diverse and comprehensive abstraction.

Model	Method	Cross-Task				Cross-Website				Cross-Domain			
		EA	AF	Step SR	SR	EA	AF	Step SR	SR	EA	AF	Step SR	SR
GPT-4o-mini	MindAct	33.9	42.8	29.8	0.0	27.5	35.6	22.6	0.0	29.3	37.1	26.1	0.0
	AWM	33.5	42.6	28.8	0.0	28.1	34.6	22.1	0.0	29.7	38.7	26.9	0.3
	A^2 (Ours)	**37.8**	**49.8**	**33.1**	**0.6**	**34.8**	**43.0**	**28.9**	**1.1**	**35.1**	**44.6**	**30.2**	**0.8**
Deepseek-v3	MindAct	40.9	51.7	37.0	**1.6**	34.6	42.4	28.8	0.0	34.9	44.2	**31.8**	1.0
	AWM	41.7	50.6	37.3	1.2	34.3	41.5	28.2	0.0	34.3	42.5	30.8	**1.2**
	A^2 (Ours)	**43.7**	**52.4**	**37.7**	0.8	**37.3**	**43.6**	**30.0**	**0.6**	**36.3**	**44.5**	30.5	**1.2**

Table 1: Evaluation performance (%) on Mind2web benchmark using GPT-4o-mini and deepseek-v3 under the zero-shot setting. EA is short for element accuracy and AF is short for action F1 score. SR is short for success rate.

Method	Shopping	CMS	Reddit	Gitlab	Avg.
TaE	17.2	17.6	14.0	3.6	13.1
AWM	17.7	14.8	14.0	4.6	12.8
A^2 (Ours)	**24.0**	**19.8**	**19.3**	**7.7**	**17.7**

Table 2: Success rate (%) on Webarena benchmark using GPT-4o-mini under the zero-shot setting.

Given the diversity of domains and task contexts, providing annotated examples for each task class, such as one-shot or few-shot settings, is neither feasible nor practical for evaluating the reasoning process. In real-world applications, models are expected to operate in zero-shot settings, which better assess their ability to tackle complex web tasks. Based on these considerations, we adopt a **zero-shot** reasoning approach in our experiments. While our method does not rely on annotated examples for reasoning, we involve a single fixed example to ensure format correctness during abstraction derivation in our approach and workflow generation in baseline methods. The example remains consistent throughout the entire experiment.

5.2 Main Results

As shown in Table 1 and Table 2, A^2 achieves significant performance improvements across all datasets. On the Webarena dataset, our approach outperforms baseline methods with an average success rate improvement of 4.6%. On the Mind2web dataset, our approach achieves average improvements of 5.5% in element accuracy (EA), 7.2% in action F1 (AF), 4.5% in step success rate (Step SR), and 0.7% in overall success rate (SR) separately using GPT-4o-mini. The consistent improvements across all subsets validate the superiority of our method. Notably, on the cross-website subset of the Mind2web dataset, our approach significantly improves performance, increasing EA from 28.1% to 34.8%, AF from 35.6% to 43.0%, Step SR from 22.6% to 28.9%, and SR from 0.0% to 1.1%. Additionally, A^2 exhibits superior performance with deepseek-v3 model. To further analyze the results, we provide the following detailed observations.

A^2 performs well on unfamiliar tasks, generalizing effectively to unseen data. The Mind2web dataset comprises three subsets designed to evaluate an agent's ability to generalize across domains, websites, and tasks. A detailed introduction to these subsets is provided in the appendix A. Among these, the cross-task subset closely resembles the demonstration tasks in the training set, while the cross-website and cross-domain subsets contain more unfamiliar tasks, serving to assess agents' reasoning ability in novel environments. Compared to other baseline methods, A^2 demonstrates greater performance improvements on the cross-website and cross-domain subsets relative to the cross-task subset. The potential reason may be that summarizing experiences assists in solving unfamiliar contexts and tasks, whereas previously encountered tasks rely heavily on knowledge acquired during the models' training process.

Performance varies with task distribution and dataset diversity. The improvement achieved by A^2 across different datasets on both Mind2web and Webarena is not directly proportional to the number of tasks in a subset. Instead, performance is influenced by task distribution and similarity. When tasks are sparsely distributed, the improvement tends to be less pronounced. For example, in the cross-task subset, which exhibits high task diversity (as shown in Table 5 in the appendix), the performance gain is relatively smaller compared to others. These results show that task distribution and diversity are key to overall reasoning performance.

5.3 Ablation Study

We further investigate the effectiveness of the semantic and episodic abstraction in A^2, focusing on their individual contributions by sequentially removing each component. Table 3 presents the

Semantic Abs.	Episodic Abs.	EA	AF	Step SR	SR
✗	✗	27.5	35.6	22.6	0.0
✔	✗	33.6	42.8	22.7	0.0
✔	✔	**34.8**	**43.0**	**28.9**	**1.1**

Table 3: Ablation performance (%) on test_website from the Mind2web benchmark. Abs. stands for abstraction.

Method (Sequence)	EA	AF	Step SR	SR
MindAct (Original)	27.5	35.6	22.6	0.0
AWM (Original)	28.1	34.6	22.1	0.0
A^2 (Random)	**35.2**	41.9	27.8	**1.1**
A^2 (Easiest2hardest)	32.1	42.7	26.9	0.0
A^2 (Hardest2easiest)	33.5	42.4	27.0	0.6
A^2 (Original)	34.8	**43.0**	**28.9**	**1.1**

Table 4: Performance (%) on test_website subset from Mind2web benchmark with different task sequences.

results of an ablation study conducted with GPT-4o-mini on the cross-website subset of the Mind2web benchmark. We observe that **incorporating both abstraction components leads to a consistent improvement**, underscoring their necessity in our approach. Specifically, semantic abstraction boosts the element accuracy and action F1 metrics, while episodic abstraction remarkably improves both step-level and overall success rates. This difference may stem from the nature of each abstraction. Semantic abstraction involves general-purpose knowledge, aiding the model in accurately identifying elements. Episodic abstraction captures task-specific patterns, reflecting the procedural and logical structure of the task. We look forward to further investigating the underlying reasons in future research.

5.4 Task Sequence Sensitivity Analysis

The abstraction knowledge is accumulated through the sequential task-solving process, making the task sequence a crucial factor influencing overall performance. We investigate the impact under four distinct task sequence settings: (1) the original order, (2) random shuffling, (3) from easiest to hardest, and (4) from hardest to easiest (where the number of actions in the ground truth measures task complexity). Table 4 presents the results of these task sequences with GPT-4o-mini on the Mind2web cross-website dataset. For comparison, baseline scores are reported under the original task sequence. The results indicate that while task sequence has a minor impact on reasoning performance, A^2 consistently outperforms other baselines across all four metrics in every sequence setting. This highlights the robustness and superiority of our approach A^2.

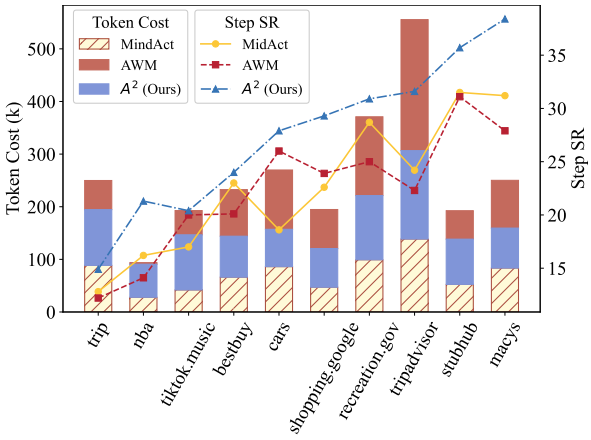

Figure 4: Token cost and step success rate (Step SR) on 10 websites from the Mind2web cross-website dataset.

5.5 Computational Efficiency Analysis

We evaluate the token costs of different methods on the Mind2web cross-website dataset using GPT-4o-mini, as illustrated in Fig. 4. A^2 significantly reduces token usage compared to previous workflow summarization methods like AWM. Specifically, our approach updates semantic abstraction by incorporating the latest task-solution pair, deriving task-adaptive episodic abstraction with comparatively limited length to steer subsequent reasoning. In contrast, AWM constructs lengthy workflows by integrating all previously solved task-solution pairs, leading to substantially higher token expenditure. Furthermore, A^2 achieves higher step success rates (Step SR) across all 10 websites compared to both MindAct and AWM. These results underscore that our approach not only delivers superior performance but also maintains optimal cost-efficiency.

6 Conclusion

In this paper, we propose A^2, a novel reasoning framework that integrates task-adaptive hierarchical abstraction from experience to enhance web task reasoning. Our approach extracts general-purpose semantic abstraction from experience and dynamically adapts it into episodic abstraction based on the next task query. By leveraging episodic abstraction to guide reasoning, our method effectively filters out irrelevant details and misleading knowledge, improving both accuracy and efficiency. A^2 achieves state-of-the-art performance while maintaining high cost-effectiveness. We encourage further research on abstraction-based reasoning to advance the effectiveness of LLMs in successfully learning abstraction knowledge from experiential data to fit in dynamic task contexts.

Limitations

In this paper, we propose A^2, a novel framework to assist reasoning with hierarchical abstractions. Despite the promising performance of our approach, we acknowledge that there are areas for improvement and opportunities for future research.

First, our evaluation is primarily conducted on complex web task datasets, which concentrate on instruction execution in dynamic and context-rich website environments. While this highlights the approach's ability to handle complex reasoning, it does not assess broader reasoning capabilities, such as algebra calculation or program synthesis. Extending the evaluation to diverse reasoning tasks encompassing these abilities would better demonstrate the generalization of our approach.

Second, we currently focus on the construction and leveraging of task-adaptive hierarchical abstraction of experiential data. However, potential knowledge conflicts may arise between the derived abstraction knowledge and subsequent task queries, further influencing the subsequent reasoning performance. Investigating methods to detect and resolve such conflicts in the abstraction process represents a valuable direction for future research.

Ethical Considerations

This article adheres to the ACL Code of Ethics. According to our knowledge, our work constitutes foundational research, and we do not identify any significant risks related to malicious harm, environmental impact, fairness, or privacy concerns.

Aknowledgements

This work was supported by the National Science and Technology Major Project (No. 2022ZD0114903) and the Natural Science Fundation of China (NSFC. No. 62476149).

References

John R Anderson. 2013. *The adaptive character of thought*. Psychology Press.

Matthew Bowers, Theo X. Olausson, Lionel Wong, Gabriel Grand, Joshua B. Tenenbaum, Kevin Ellis, and Armando Solar-Lezama. 2023. Top-down synthesis for library learning. *Proc. ACM Program. Lang.*, 7(POPL):1182–1213.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

François Chollet. 2019. On the measure of intelligence. *CoRR*, abs/1911.01547.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samual Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, and Zhifang Sui. 2024. A survey on in-context learning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 1107–1128. Association for Computational Linguistics.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. 2024. WorkArena: How capable are web agents at solving common knowledge work tasks? In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 11642–11662. PMLR.

Kevin Ellis, Catherine Wong, Maxwell I. Nye, Mathias Sablé-Meyer, Luc Cary, Lucas Morales, Luke B. Hewitt, Armando Solar-Lezama, and Joshua B. Tenenbaum. 2020. Dreamcoder: Growing generalizable, interpretable knowledge with wake-sleep bayesian program learning. *CoRR*, abs/2006.08381.

Kit Fine. 2002. *The limits of abstraction*. Clarendon Press.

Peter Conway Humphreys, David Raposo, Tobias Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Adam Santoro, and Timothy P. Lillicrap. 2022. A data-driven approach for learning to control computers. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 9466–9482. PMLR.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi

Deng, Chenyu Zhang, Chong Ruan, et al. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. 2018. Reinforcement learning on web interfaces using workflow-guided exploration. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024b. Lost in the middle: How language models use long contexts. *Trans. Assoc. Comput. Linguistics*, 12:157–173.

Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. 2023. Large language models as general pattern machines. In *Conference on Robot Learning, CoRL 2023, 6-9 November 2023, Atlanta, GA, USA*, volume 229 of *Proceedings of Machine Learning Research*, pages 2498–2518. PMLR.

Michael C Mitchelmore. 2002. The role of abstraction and generalisation in the development of mathematical knowledge.

Arsenii Moskvichev, Victor Vikram Odouard, and Melanie Mitchell. 2023. The conceptarc benchmark: Evaluating understanding and generalization in the ARC domain. *Trans. Mach. Learn. Res.*, 2023.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

OpenAI. 2024. Gpt-4o mini: advancing cost-efficient intelligence. `https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/`.

Jean Piaget. 1977. *La naissance de l'intelligence chez l'enfant*. FeniXX.

Jean Piaget. 2013. *The construction of reality in the child*. Routledge.

Yisheng Song, Ting Wang, Puyu Cai, Subrota K. Mondal, and Jyoti Prakash Sahoo. 2023. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. *ACM Comput. Surv.*, 55(13s):271:1–271:40.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. 2024. Agent workflow memory. *CoRR*, abs/2409.07429.

Fangzhi Xu, Qika Lin, Jiawei Han, Tianzhe Zhao, Jun Liu, and Erik Cambria. 2023. Are large language models really good logical reasoners? A comprehensive evaluation from deductive, inductive and abductive views. *CoRR*, abs/2306.09841.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E. Gonzalez, and Bin Cui. 2024a. Buffer of thoughts: Thought-augmented reasoning with large language models. *CoRR*, abs/2406.04271.

Zonglin Yang, Li Dong, Xinya Du, Hao Cheng, Erik Cambria, Xiaodong Liu, Jianfeng Gao, and Furu Wei. 2024b. Language models as inductive reasoners. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2024 - Volume 1: Long Papers, St. Julian's, Malta, March 17-22, 2024*, pages 209–225. Association for Computational Linguistics.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022a. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022b. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. 2024. Is in-context learning sufficient for instruction following in llms? *CoRR*, abs/2405.19874.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. 2024. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zhaocheng Zhu, Yuan Xue, Xinyun Chen, Denny Zhou, Jian Tang, Dale Schuurmans, and Hanjun Dai. 2023. Large language models can learn rules. *CoRR*, abs/2310.07064.

A Datasets

In this section, we provide a detailed introduction to the two web task datasets utilized in the main experiments, *i.e.*, Mind2web and Webarena.

Mind2web is a novel dataset containing natural language task descriptions and manually annotated action sequences. It contains 5 top-level domains: Travel, Shopping, Service, Entertainment, and Information. Subsequently, these are separated into 31 secondary domains. For each domain, 3-5 websites are manually chosen, resulting in 137 websites in total. The statistics are shown in Table 5. For the task proposal stage, to boost creativity and diversity, they use ChatGPT to propose seed tasks, based on which annotators are required to generate various tasks. During the task demonstration period, workers use a Playwright-based tool to demonstrate how to perform the proposed tasks in a web browser. The workers subsequently execute element selection and operation selection to ensure accuracy. Mind2web contains 2,350 tasks, 1,135 elements on average, and 7.3 actions per task on average.

Webarena is a realistic and reproducible web environment designed to facilitate the development of autonomous agents to execute tasks. It contains four self-host domains, *i.e.*, online shopping, discussion forums, collaborative development, and business content management. The dataset contains 812 tasks, each with a natural language intent, emulating the abstract language usage patterns typically employed by humans. The evaluation focuses on functional correctness, which measures whether the actions actually solve the task goal. The benchmark contains 241 templates and 812 instantiated intents. On average, each template is instantiated to 3.3 examples. The intents can be roughly divided into three classes, *i.e.*, information seeking, site navigation, and configuration operation.

B Implement Details

B.1 Evaluation

For Mind2web, we follow MindAct to compare the reasoning results with the ground truth, measuring the four metrics: element accuracy, action F1 score, step success rate, and overall success rate. For the Webarena dataset, there is no golden ground truth for evaluation. The benchmark designs an evaluation process to measure the extent to which the actions satisfy the task. We follow previous techniques to evaluate using an LLM, which is kept the same as the corresponding reasoning model.

Subset	#Website	#Task	Avg. Task	#D	#Sub
test_task	69	252	3.65	3	17
test_website	10	177	17.7	3	9
test_domain	54	912	16.89	2	13

Table 5: Min2web statistics, including the number of websites, number of tasks, average tasks per website, number of domains, and number of subdomains.

B.2 Models

Web task reasoning requires long contexts because of the abundant options and dynamic web states. Therefore, models that only support small dialogue context lengths may fail to reason effectively, leading to incorrect evaluation. Based on the above consideration and price limit, we use GPT-4o-mini as our backbone model. We use deekseek-v3 to facilitate comparison. The experiments are mainly completed in Dec 2024. The reasoning and evaluation models are kept the same in all experiments. We would like to emphasize that the models for assimilation and accommodation are not required to be the same. In our experiments, the models are kept the same for a fair comparison. However, it is important to note that this is not a strict requirement: our approach allows for flexibility in choosing either the same or different models for these steps, depending on implementation needs.

B.3 Parameters

To accommodate the long contexts required for solving web tasks, we set the maximum token limit to 128,000 for the WebArena dataset, which is uniformly applied across all methods. To ensure reproducibility, we consistently set the temperature to zero, except when evaluating the WebArena dataset. Following Wang et al. (2024), we set the temperature to 1.0 in this case to enhance output diversity.

As discussed in Sec. 5, it is impractical nor efficient to manually design demonstrations for each website. Therefore, we consistently adapt zero-shot settings through our experiments. For Mind2web, the number of positive candidate elements is kept at 5, following the default settings of MindAct.

For the Webarena dataset, we follow AWM to utilize Browsergym for our experiments. Browsergym is a framework designed to address the growing need for efficient evaluation and benchmarking of web agents. It involves various existing benchmarks for comprehensive evaluations. Browsergym aims to boost the evaluation of web agents by providing a unified, gym-like environment with well-defined observation and action spaces. By

Method	$\sigma \pm \mu$	CV
EA for AWM	26.9750±1.03188	0.03767
EA for A^2	33.9000±1.4750	**0.0358**
AF for AWM	35.0000±0.3250	0.0163
AF for A^2	42.5000±0.1650	**0.0096**
Step SR for AWM	22.3500±0.5675	0.0337
Step SR for A^2	27.6500±0.6425	**0.0290**

Table 6: Robustness comparison between A^2 and AWM.

Website/Metric	EA	AF	Step SR
shopping.google	0.01695	0.00009	0.02401
trip	0.00450	0.00301	0.01641
recreation.gov	0.01132	0.00453	0.00130
tripadvisor	0.00001	0.00073	0.00003
cars	0.02164	0.01436	0.00440
tiktok.music	0.00036	0.00426	0.00068
stubhub	0.03581	0.03726	0.01806
nba	0.01013	0.00146	0.00300
macys	0.00890	0.00634	0.00763
bestbuy	0.00346	0.00005	0.01190

Table 7: P-values for A^2 and MindAct across different metrics and websites.

standardizing the evaluation, the framework is able to reduce the time cost as well as the complexity of developing web agents.

B.4 Model Prompts

We provide model prompts in Listing 1 to enhance reproducibility. The prompts mainly contain three key components as follows.

- **System task description:** Defines the objective-updating semantic abstraction during assimilation and specifies the expected output abstraction format.
- **Existing semantic abstraction:** The abstraction follows a hierarchical structure implemented through numbering, special tokens, and indentation. Each entry includes a natural language task description and concrete actions. Similar to Python, indentation and finer-grained numbering indicate task relationships. Subroutines are described, with execution steps in backticks and dynamic parameters in curly braces (e.g., web element IDs). This ensures a clear, flexible structure.
- **One demonstration example:** A fixed example ensures correct formatting and minimizes manual effort by remaining consistent.

For further details, refer to our GitHub repository: https://github.com/Xinyu-Pang/A2.

C Further Analysis

C.1 Robustness

To provide a more comprehensive analysis, we expanded the experiments on the Mind2web cross-website subset, comparing A^2 and AWM under four task ordering settings: Random, Easiest2hardest, Hardest2easiest, and Original. We report the mean (μ), standard deviation (σ), and coefficient of variation (CV) to assess relative performance changes, where $CV = \mu/\sigma$. Since MindAct does not incorporate prior task information, its performance remains stable across different task orders and is thus not relevant for this comparison. Additionally, as AWM consistently achieves an SR of 0.0, we focus on the remaining metrics: EA, AF, and Step SR. The results are represented in Table 6.

The results indicate that A^2 consistently achieves lower CV and higher average performance across different metrics. This suggests that our approach remains robust and consistently outperforms baseline methods across different task order settings.

C.2 T-test

To further confirm the statistical significance of our findings, we conducted a t-test comparing A^2 with MindAct. For this analysis, we selected the cross-website subset from Mind2web, which contains 10 websites. GPT-4o-mini is used as the base model, and the dataset is randomly shuffled 5 times. Both A^2 and MindAct are evaluated separately on these shuffled datasets. The evaluation metrics included element accuracy (EA), action F1 (AF), and step success rate (Step SR). All other experimental settings were maintained following the previous experiments. The t-test was performed using the scipy Python package, and the p-values for different metrics and websites are shown in Table 7.

The hypothesis tested was: "A^2 achieves significantly higher scores (EA, AF, or Step SR) than MindAct." Across all 10 websites, the t-test consistently yields low p-values across every metric, ranging from 0.00001 to 0.03726. Using the standard significance threshold of 0.05, we confirm that the hypothesis holds: A^2 significantly outperforms MindAct across every evaluation metric. Combining the above t-test results and the reasoning performance, we concluded that A^2 achieves superior performance over baseline approaches.

```
# Python prompt for deriving semantic abstraction.


**System Prompt**
<Task> Integrate the latest solved task (q_i) and next task query (q_{i+1}) with
     the existing abstraction (a_{i-1}) to refine a new abstraction (a_i). The
    new abstraction contains information from all previous tasks and is
    hierarchically organized. (a_i) should assistant in solving (q_{i+1}).

<Abstraction Requirements>
1. Hierarchical: The abstractions should be multi-layered. Higher-level
    abstractions should represent lower-level ones while ignoring irrelevant
    details. The structure and layer depth should be dynamically adjusted.
2. Comprehensive: (a_i) should encapsulate patterns from all tasks, balancing
    task-specific details (in lower-level abstractions) and general principles (
    in higher-level abstractions).

<Task Format> The extracted abstractions should follow this format: {
    Operation_type} [{element_id}] {Operation_value} ([{element_type} {
    element_content} -> {Operation_type}]). If the Operation_type is CLICK, then
     {Operation_value} should not be included.

**1-shot Demonstration Input**
## Query 1: Book a cheapest bundle and save option for 2 adults from Ahmedabad
    to Dubai on April 5 with free cancellation options, hotel should be a 3 star
     near Burj Khalifa with guest rating above 4, one night.
    Actions:
        `Action: [span]  Bundle & Save -> CLICK`
        `Action: [textbox]  From -> TYPE: ahmedabad`
        `Action: [b]  Ahmedabad -> CLICK`
        `Action: [textbox]  To -> TYPE: dubai`
        `Action: [span]  DXB -> CLICK`
        `Action: [textbox]  Depart -> CLICK`
        `Action: [listitem]  5 -> CLICK`
        `Action: [listitem]  6 -> CLICK`
        `Action: [div]  Search -> CLICK`
        `Action: [span]  4+ -> CLICK`
        `Action: [div]  Price (Low to High) -> CLICK`
        `Action: [button]  Book -> CLICK`

<Split token>
1. Choose task type
    #{task} could be Trains, Flights, and so on, depending on the task
        description and html
    `Action: CLICK [{element_id}] ([span] {task} -> CLICK)`

2. Add constraints
    2.1 Set location
        `Action: CLICK [{element_id}] ([{element_type}] City or Region -> CLICK)
            `
        `Action: TYPE [{element_id}] {location} ([{element_type}] {City or
            airport} -> TYPE: {location})`
    2.2 Set time
        `Action: CLICK [{element_id}] ([{element_type}] {arrive_date} -> CLICK)`
        `Action: CLICK [{element_id}] ([{element_type}] {leave_date} -> CLICK)`

3. Action
    # {action} could be Search, Select, and so on.
    `Action: CLICK [{element_id}] ([{element_type}] {action} -> CLICK)`

4. Other
    `Action: CLICK [{element_id}] ([i]   -> CLICK)
```

Listing 1: Prompt for deriving semantic abstraction (part 1).

```
**1-shot Demonstration Output**

<Split token>
1. Choose task type
    #{task} could be Trains, Flights, Bundle&Save, and so on, depending on the
        task description and html
    `Action: CLICK [{element_id}] ([span] {task} -> CLICK)`

2. Add constraints
    2.1 Set location
        2.1.1 Choose city or airport
            `Action: CLICK [{element_id}] ([{element_type}] City or Region ->
               CLICK)`
            `Action: TYPE [{element_id}] {location} ([{element_type}] {City or
               airport} -> TYPE: {location})`
        2.1.2 Type in locations
            `Action: [textbox]  From -> TYPE: {starting_origin}`
            `Action: [b]  {starting_origin} -> CLICK`
            `Action: [textbox]  To -> TYPE: {destination}`
            `Action: [span]  {destination} -> CLICK`

    2.2 Set time
        `Action: CLICK [{element_id}] ([{element_type}] {arrive_date} -> CLICK)`
        `Action: CLICK [{element_id}] ([{element_type}] {leave_date} -> CLICK)`

3. Action
    # {action} could be Search, Book, Select, and so on.
    `Action: CLICK [{element_id}] ([{element_type}] {action} -> CLICK)`

4. Sort
    4.1 Sort by price (from low to high)
        `Action: CLICK [{element_id}] ([div]  Price (Low to High) -> CLICK)`

**Task Input**
## Query: {specific task query}
    Actions: {detailed action sequences}
## Existing Abstraction: {exsiting semantic abstraction}
```

Listing 2: Prompt for deriving semantic abstraction (part 2).

```
# Python prompt for deriving episodic abstraction.

**System Prompt**
<Task> Integrate the existing semantic abstraction (a_{i}) and next query (q_{i
    +1}) to refine a new abstraction (\hat{a_i}). The new abstraction contains
    useful information that assists in solving nthe ext query.

<Abstraction Requirements>
1. Hierarchical: The abstractions should be multi-layered. Higher-level
    abstractions should represent lower-level ones while ignoring irrelevant
    details. The structure and layer depth should be dynamically adjusted.
2. Comprehensive: (a_i) should encapsulate patterns from all tasks, balancing
    task-specific details (in lower-level abstractions) and general principles (
    in higher-level abstractions).

<Task Format> The extracted abstractions should follow this format: {
    Operation_type} [{element_id}] {Operation_value} ([{element_type} {
    element_content} -> {Operation_type}]). If the Operation_type is CLICK, then
     {Operation_value} should not be included.
```

Listing 3: Prompt for deriving episodic abstraction (part 1).

```
**1-shot Demonstration Input**
## Query 2: Select a high speed train ticket with a departure time before 23:00
     from Shanghai to Beijing.

<Split token>
1. Choose task type
    #{task} could be Trains, Flights, Bundle&Save, and so on, depending on the
        task description and html
    `Action: CLICK [{element_id}] ([span] {task} -> CLICK)`

2. Add constraints
    2.1 Set location
        2.1.1 Choose city or airport
            `Action: CLICK [{element_id}] ([{element_type}] City or Region ->
               CLICK)`
            `Action: TYPE [{element_id}] {location} ([{element_type}] {City or
               airport} -> TYPE: {location})`
        2.1.2 Type in locations
            `Action: [textbox]  From -> TYPE: {starting_origin}`
            `Action: [b]  {starting_origin} -> CLICK`
            `Action: [textbox]  To -> TYPE: {destination}`
            `Action: [span]  {destination} -> CLICK`

    2.2 Set time
        `Action: CLICK [{element_id}] ([{element_type}] {arrive_date} -> CLICK)`
        `Action: CLICK [{element_id}] ([{element_type}] {leave_date} -> CLICK)`

3. Action
    # {action} could be Search, Book, Select, and so on.
    `Action: CLICK [{element_id}] ([{element_type}] {action} -> CLICK)`

4. Sort
    4.1 Sort by price (from low to high)
        `Action: CLICK [{element_id}] ([div]  Price (Low to High) -> CLICK)`

**1-shot Demonstration Output**
<Split token>
1. Choose task type to book train tickets
    `Action: CLICK [{element_id}] ([span] {Trains} -> CLICK)`

2. Add constraints
    2.1 Type in locations
        `Action: [textbox]  From -> TYPE: {shanghai}`
        `Action: [b]  {shanghai} -> CLICK`
        `Action: [textbox]  To -> TYPE: {beijing}`
        `Action: [span]  {beijing} -> CLICK`

    2.2 Set departure time
        `Action: CLICK [{element_id}] ([{element_type}] {23:30} -> CLICK)`

3. Book action
    # {action} could be Search, Book, Select, and so on.
    `Action: CLICK [{element_id}] ([{element_type}] {action} -> CLICK)`

**Task Input**
## Query: {next task query}
## Existing Abstraction: {exsiting episodic abstraction}
```

Listing 4: Prompt for deriving episodic abstraction (part 2).