RetailBench: Evaluating Long-Horizon Autonomous Decision-Making and Strategy Stability of LLM Agents in Realistic Retail Environments

Anonymous ACL submission

Abstract

Large Language Model (LLM)-based agents have achieved notable success on short-horizon and highly structured tasks, yet their ability to maintain coherent decision-making over long horizons in dynamic environments remains an open challenge. We introduce *RetailBench*, a high-fidelity benchmark designed to evaluate long-horizon autonomous decision-making in realistic commercial scenarios, where agents must operate under stochastic demand and evolving external conditions.

We further propose the *Evolving Strategy & Execution* framework, which separates high-level strategic reasoning from low-level action execution, enabling adaptive and interpretable strategy evolution over time. This design is crucial for long-horizon tasks, where non-stationary environments and error accumulation require strategies to be revised at a different temporal scale than action execution. Experiments on seven state-of-the-art LLMs across progressively challenging environments show that our framework improves operational stability and efficiency compared to a Reflection-based baseline. However, performance degrades substantially as task complexity increases, revealing fundamental limitations in current LLMs for long-horizon, multi-factor decision-making.

1 Introduction

Recent large language models (LLMs), particularly when augmented with reasoning and tool-use capabilities, have demonstrated strong performance on a variety of cognitively demanding tasks, including code editing, mathematical problem solving, and complex information retrieval (Jimenez et al., 2024; Phan et al., 2025; Gao et al., 2024). However, growing evidence indicates that these capabilities do not readily translate into robust, general-purpose autonomy, especially in settings that require long-term planning, persistent goal maintenance, and adaptation to dynamic environments (Amodei, 2024; Kwa et al., 2025; METR, 2025). Correspondingly, existing agent benchmarks—spanning web interaction (Mialon et al., 2023; Wei et al., 2025; Zhou et al., 2024; Deng et al., 2023; Jimenez et al., 2024; Team, 2025b) primarily focus on short-horizon or highly structured tasks, limiting their ability to evaluate sustained interaction with complex, evolving environments. Recent studies on long-horizon autonomy consistently demonstrate that even state-of-the-art agents struggle to maintain coherent strategies over extended time spans (Nof1.ai, 2025; Andon Labs, 2025; Backlund and Petersson, 2025).

To systematically study this challenge, we introduce *RetailBench*, a new benchmark grounded in real-world commercial data and informed by established economic modeling principles.RetailBench centers on a supermarket operation scenario that demands long-horizon decision-making, sustained interaction with a dynamic environment, and the integration of heterogeneous historical information. The benchmark evaluates whether LLM-based agents can autonomously sustain realistic business operations under complex, multi-factor conditions.

We evaluate seven state-of-the-art LLMs in this environment. Our results show that current models struggle to maintain stable decision quality as the decision space expands and often fail to incorporate all relevant information. Moreover, hallucinations and economically irrational behaviors frequently emerge during long-horizon execution, leading to environment collapse and preventing sustained autonomous operation.

Our contributions are summarized as follows:

- We introduce *RetailBench*, a high-fidelity benchmark for evaluating long-horizon autonomous decision-making in realistic retail environments.
- We propose the *Evolving Strategy & Execution* agent framework, which improves operational

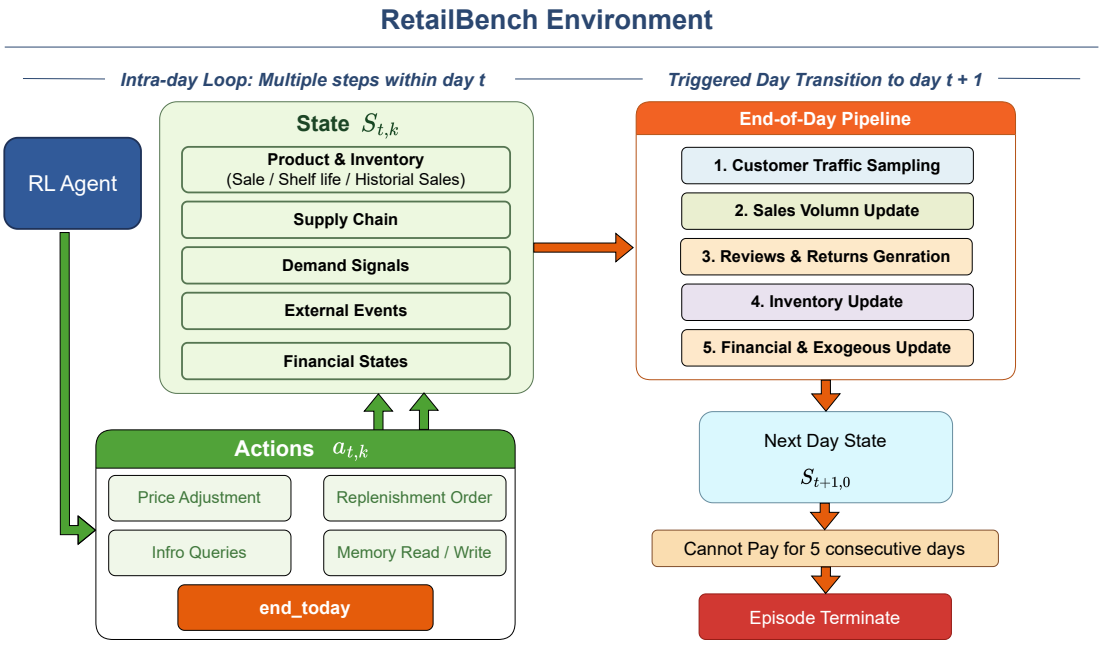

Figure 1: Overview of the hierarchical supermarket environment, illustrating intra-day agent–environment interactions and end-of-day state transition dynamics.

stability compared to a Reflection-based baseline.

- Through extensive experiments, we identify systematic failure modes of current LLM-based agents in long-horizon, multi-factor decision-making settings.

2 Environment Construction

2.1 Problem Formulation and Overview

We model supermarket operations as a Markov Decision Process (MDP), in which an autonomous agent manages a single retail store over a finite horizon of days. At each day t, the agent makes a sequence of operational decisions that jointly determine the store's daily outcomes.

Formally, the MDP is defined as $(\mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, $\mathcal{T}$ the stochastic transition dynamics, R the reward function, and $\gamma \in (0, 1]$ the discount factor.

At the beginning of each day t, the agent observes the initial state $S_{t,0}$ and executes a sequence of intra-day actions indexed by k. Each action $a_{t,k} \in \mathcal{A}$ induces a transition to the next intra-day state $S_{t,k+1}$. When the agent issues the *end-today* action, the environment transitions to the initial state of the next day, $S_{t+1,0}$, according to the transition dynamics $\mathcal{T}$. Figure 1 illustrates the overall interaction process.

The environment supports long-horizon operation over more than one thousand simulated days, with episodes terminating if the store fails to pay rent for five consecutive days.

2.2 State Space

The intra-day state $S_{t,k}$ summarizes the complete operational context of the store at step k of day t and is composed of multiple interdependent components: $S_{t,k} = \left(S^{\text{prod}}_{t,k}, S^{\text{inv}}_{t,k}, S^{\text{sup}}_{t,k}, S^{\text{dem}}_{t,k}, S^{\text{ext}}_{t,k}, S^{\text{fin}}_{t,k}\right)$.

- $S^{\text{prod}}_{t,k}$ and $S^{\text{inv}}_{t,k}$ encode product-level attributes and on-hand inventory status, including prices, shelf life, and historical sales records. Product demand is grounded in real-world retail data derived from the Dominick's dataset (Kilts Center for Marketing).

- $S^{\text{sup}}_{t,k}$ represents the supply chain state, including supplier prices, quality levels, and delivery lead times, constructed to reflect empirically observed price–quality relationships (Grewal et al., 2014).

- $S^{\text{dem}}_{t,k}$ captures demand-side signals such as recent customer traffic and aggregated review statistics, which influence consumer purchasing behavior (Fedewa et al., 2021).

- $S^{\text{ext}}_{t,k}$ represents external contextual information, including active news events with market-,

category-, or product-level impact scope, synthesized to resemble real-world financial news (ashraq, 2025).

- $S^{\text{fin}}_{t,k}$ denotes the financial state of the store, including available cash and estimated net worth, accounting for inventory depreciation over product shelf life.

Together, these components define a richly structured yet partially stochastic state space. Further details of the environment design are deferred to Appendix A.

2.3 Action Space

At each time step t, the agent selects an action $a_t \in \mathcal{A}$, defined as a tuple of decision components: $a_t = \left(a_t^{\text{price}}, a_t^{\text{repl}}, a_t^{\text{info}}, a_t^{\text{mem}}, a_t^{\text{end}}\right)$. Each component corresponds to a distinct class of operational decisions:

- a_t^{price} denotes pricing decisions for individual SKUs;
- a_t^{repl} denotes inventory replenishment decisions, including supplier selection and order quantities;
- a_t^{info} denotes information acquisition actions, such as querying sales history, customer reviews, supplier conditions, or current news;
- a_t^{mem} denotes memory operations that allow the agent to write and retrieve persistent notes across days;
- a_t^{end} denotes a day-termination action that concludes the current decision phase.

Within a single day, the agent may execute multiple information queries and operational adjustments before issuing the a_t^{end} action to advance the environment to the next day.

2.4 Day Transition Dynamics

After the agent issues the day-termination action, the environment transitions to the next day according to $S_{t,k+1} \sim \mathcal{T}(\cdot \mid S_{t,k}, a_t)$, where the transition function $\mathcal{T}$ factorizes into a sequence of stochastic updates that jointly model daily supermarket operations.

Specifically, each day-level transition consists of the following steps:

1. **Customer traffic sampling.** The total customer traffic for the day is sampled according to stochastic demand dynamics.
2. **Sales realization.** Given the sampled customer traffic and exogenous factors (e.g., promotions, seasonal effects, and news signals), the realized sales volume of each product is determined.
3. **Reviews and returns generation.** Customer reviews and product return events are generated conditional on sales outcomes, product quality, and external influences.
4. **Inventory update.** Sold units are deducted from on-hand inventory, and replenishment orders scheduled to arrive on the current day are added to stock.
5. **Financial and exogenous state update.** The agent's financial state is updated based on realized revenues and costs, after which new exogenous information (e.g., news or market signals) for the next day is generated.

3 Evolving Strategy & Execution Framework

3.1 Framework Detail

We propose an explicit two-stage interaction framework, termed *Evolving Strategy & Execution*, for modeling strategy formation and execution in long-horizon interactive decision-making tasks. The central design principle is to decouple high-level strategic reasoning from low-level action execution, thereby making the agent's strategy explicit, analyzable, and temporally persistent throughout interaction.

In the *Evolving Strategy* stage, the agent may invoke observation and analysis tools to inspect environmental feedback and historical outcomes, but is prohibited from taking actions that directly alter the environment. Based on these observations, the agent decides whether and how to revise the strategy inherited from previous steps. Strategy updates are exploratory and may involve adding, modifying, or removing strategy components. This stage is purely deliberative, ensuring that long-term intent and planning logic are made explicit rather than implicitly entangled with immediate action choices.

In the *Execution* stage, the current strategy is frozen and treated as fixed. The agent generates concrete actions strictly in accordance with this strategy, without further strategy modification. This separation enables clearer attribution of behavioral outcomes to the underlying strategy.

Model	Avg. Days ↑	Avg. Daily Sales ↑	Avg. Daily Income ↑	Expiry Ratio ↓	Return Ratio ↓	Max Days ↑
Framework: Evolving Strategy & Execution						
DeepSeek-V3.2 (Exp.)	58.33	229.19	183.26	0.0889	0.1122	66
Gemini-3 (Fast)	50.67	399.39	294.71	0.0799	0.1311	59
GLM-4.6	52.40	174.34	124.67	0.0773	0.1293	58
Grok-4.1 Fast	**61.75**	**508.08**	**336.94**	0.0417	**0.0847**	**88**
Kimi-K2 (Thinking)	54.25	260.68	168.72	**0.0239**	0.1179	58
OpenAI-5.1 Mini	51.75	192.90	122.46	0.0360	0.1237	55
Qwen-235B (Thinking)	37.50	420.31	236.50	0.0745	0.0852	48
Average (7 models)	**52.38**	**301.98**	**203.30**	**0.0590**	**0.1068**	**61.71**
Framework: Reflection						
DeepSeek-V3.2(Exp.)	53.00	235.01	170.04	0.0382	0.1200	66
Gemini-3 (Fast)	45.67	**447.74**	**255.38**	0.0682	0.1350	50
GLM-4.6	55.00	160.70	125.67	**0.0194**	0.1176	62
Grok-4.1 Fast	48.33	297.94	197.54	0.1460	**0.0925**	54
Kimi-K2 (Thinking)	**58.33**	216.51	184.01	0.0964	0.1255	**71**
OpenAI-5.1 Mini	53.33	93.04	92.11	0.1062	0.1331	59
Qwen-235B (Thinking)	37.00	420.77	202.79	0.2645	0.1551	43
Average (7 models)	50.10	248.39	170.88	0.1092	0.1249	57.86
Heuristic Policy (Upper Bound, Easy)						
Hand-crafted Policy	180.00	674.18	729.46	0.0266	0.007	180

Table 1: Performance comparison of seven large language models under different agent frameworks in the EASY environment. A hand-crafted heuristic policy is included as an approximate upper bound.

By alternating between these two stages, the proposed framework enforces a principled separation between strategic deliberation and operational execution. This design mitigates uncontrolled strategy drift, promotes behavioral stability over long horizons, and facilitates more interpretable analysis of agent decision-making dynamics in complex environments. An illustration is provided in Appendix D.1.

3.2 Hierarchical Policy Representation

To support structured, interpretable, and temporally extended decision-making under the proposed framework, we represent the agent policy using a hierarchical abstraction that separates strategic intent from executable actions. Each policy consists of three conceptual layers:

1. **Macro Strategy**, which captures high-level managerial objectives that persist across multiple decision steps;

2. **Execution Strategy**, which encodes structured operational guidance in a machine-readable intermediate representation;

3. **Daily Actions**, which specify concrete executable operations issued to the environment.

Detailed policy configurations and example policies are provided in Appendix A.2.1.

4 Experiment Settings

We conduct experiments under three environment configurations with increasing levels of difficulty. These configurations vary in market complexity, budget constraints, and the presence of exogenous dynamics.

4.1 Environment Configurations

We employ a heuristic policy with full access to the environment's internal state as a calibration baseline for each environment variant. Environment parameters are tuned such that the heuristic policy remains stable across different difficulty levels while still experiencing meaningful operational pressure in Appendix A.4. To evaluate models' information-processing and external perception capabilities, we design three environment configurations:

- *Easy*: A controlled environment without dynamic news events or adaptive supplier price–quality relationships. The market contains five product categories. The agent is initialized with a budget of 10,000 and incurs a fixed daily rent of 250.

- *Middle*: A moderately complex environment that expands the product space to all twenty categories while still excluding dynamic news events and supplier adaptations. The initial budget is increased to 50,000, with a daily rent of 1,000.

- *Hard*: The most challenging and realistic environment, incorporating dynamically generated news events and time-varying supplier price–quality relationships. The market includes all twenty product categories. The agent starts with a budget of 50,000, pays a daily rent of 1,000, and receives twenty news items per day.

Detailed specifications of all environment configurations are provided in Appendix A.3.

4.2 Evaluation Metrics

Metrics. We evaluate store-level operational performance using the following metrics (↑ indicates higher is better; ↓ indicates lower is better):

- *Days* (↑): the number of operating days before episode termination;
- *MaxDays* (↑): the maximum number of operating days achieved across three rollouts;
- *Avg. Daily Sales* (↑): the average number of items sold per day;
- *Avg. Daily Income* (↑): the average money earned per day;
- *Expiry Ratio* (↓): the fraction of products that expire before being sold;
- *Return Ratio* (↓): the fraction of sold products that are returned by customers.

All reported metrics are averaged over three independent rollouts, each subject to a fixed maximum execution horizon.

4.3 Experimental Setup

Agent Frameworks. Preliminary experiments indicate that simple ReAct-style interaction frameworks are unstable in long-horizon settings, frequently exhibiting premature episode termination during mid-horizon execution. This motivates the evaluation of agent frameworks that explicitly support sustained strategic control.

We compare our proposed *Evolving Strategy & Execution* framework with a Reflection-based baseline (Shinn et al., 2023). Unlike our approach, the Reflection framework does not explicitly represent or preserve high-level strategies, instead relying on iterative post-hoc reflection to update a global long-term memory. Both frameworks operate under a maximum context length of 40k tokens, with each simulated day consisting of up to 50 interaction rounds; the Reflection baseline additionally performs a reflection step at the end of each day.

Full prompt specifications for both frameworks are provided in Appendix C.

Models. We conduct experiments using a diverse set of contemporary large language models, including Qwen-235B (Thinking) (Team, 2025a), Kimi K2 (Thinking) (Team et al., 2025b), GLM-4.6 (Team et al., 2025a), DeepSeek-V3.2-Exp (DeepSeek-AI et al., 2025), Gemini-3-Flash-Preview (Google DeepMind, 2024), Grok-4.1 Fast (xAI, 2025), and GPT-5-Mini (OpenAI, 2025). Due to the substantial token costs incurred during long-horizon rollouts, we use lower-cost variants for closed-source models to ensure experimental feasibility.

Evaluation Protocol. We evaluate seven large language models under each of the three environment configurations. Each model is assessed using three independent rollouts, and all reported metrics are averaged across runs. To isolate the effect of the agent framework, we further conduct a controlled comparison between the proposed framework and the Reflection baseline in the *Easy* environment, where both frameworks are evaluated under identical conditions. Finally, to quantify the gap between current LLM-based agents and an environment-optimal strategy, we implement a hand-crafted policy based on internal knowledge unavailable to the agents, which serves as an approximate upper bound on achievable performance.

5 Results

5.1 Performance Comparison between Two Agent Frameworks

Table 1 compares seven large language models in the Easy environment under two agent frameworks: *Evolving Strategy & Execution* and *Reflection*. Under the proposed framework, Grok-4.1 Fast achieves the strongest overall performance, attaining the highest average daily sales, average daily profit, and the longest maximum survival duration, indicating superior long-horizon planning and operational stability. In contrast, Kimi-K2 (Thinking) yields the lowest expiry ratio, reflecting more conservative and risk-aware inventory management.

At the framework level, *Evolving Strategy & Execution* consistently outperforms *Reflection* across all key metrics, achieving higher sales and profit

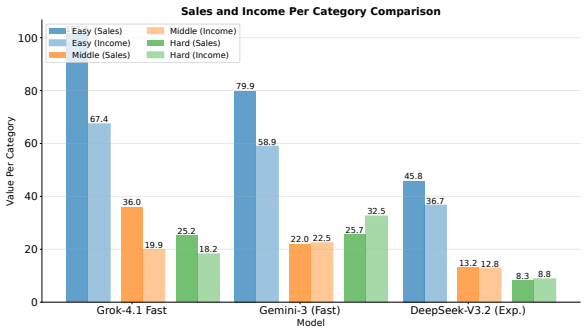

Figure 2: Category-level sales and profit per category across Easy, Middle, and Hard environments. Results are shown for three representative models.

Model	Context	Easy		Middle		Hard	
		SKU ↑	Cat ↑	SKU ↑	Cat ↑	SKU ↑	Cat ↑
DeepSeek-V3.2 (Exp.)	128K	4.75	3.31	6.83	5.34	6.64	5.45
Gemini-3 (Fast)	**1M**	**8.72**	**4.34**	**13.60**	**12.43**	**21.57**	**13.56**
GLM-4.6	200K	4.82	2.98	3.23	2.46	7.08	5.22
OpenAI-5.1 Mini	400K	3.66	1.94	4.10	4.10	6.67	4.79
Grok-4.1 Fast	200K	5.78	3.68	4.87	3.08	5.40	3.20
Kimi-K2 (Thinking)	256K	5.06	3.09	5.17	4.77	6.91	4.75
Qwen-235B (Thinking)	256K	4.59	3.67	3.17	2.36	5.11	4.51
Heuristic Strategy	–	9.03	4.87	35.34	18.61	34.82	18.52

Table 2: SKU and Category counts sold by different models across difficulties. Context denotes the maximum supported context length of each model. **Bolded** indicates the best model; underlined indicates the second best (Human excluded).

while substantially reducing product expiry. These results demonstrate that explicitly decoupling strategy evolution from execution improves both revenue efficiency and inventory control. Nevertheless, a notable performance gap remains relative to the hand-crafted heuristic policy, suggesting that current LLM-based agents still fall short of optimal long-horizon decision-making.

5.2 Performance across Environments with Varying Difficulty

As environment difficulty increases, all models exhibit consistent performance degradation, including shorter operational durations, higher expiry and return ratios, and reduced long-horizon stability. Although the expansion of product categories in the Middle and Hard settings enables higher aggregate sales and profit for some models, *Sales per Category* and *Profit per Category* decline substantially (Figure 2), indicating persistent challenges in effective resource allocation within increasingly high-dimensional decision spaces.

The relatively small performance gap between the Middle and Hard environments can be partly attributed to the delayed impact of intensified news dynamics, whose effects typically unfold over longer time horizons. Appendix B.3 reports the full results across all environments under the proposed framework. Overall, while models demonstrate limited adaptability as task complexity increases, their performance remains substantially below the heuristic upper bound, highlighting persistent limitations in robust long-horizon decision-making under complex and dynamic conditions.

6 Analysis

Based on both quantitative evaluation results and manual inspection of trajectories, we identify several key factors that explain why current models fail to operate reliably in the environment.

We analyze these failure modes from four complementary perspectives.

6.1 Non-scalable Decision-Making Capability

As shown in Appendix B.3, models achieve reasonable performance in the Easy setting but exhibit consistent degradation as the environment scales in complexity. To better understand this phenomenon, we analyze both the average number of stock keeping units (SKUs) and product categories sold per day, as well as the number of SKUs and categories explicitly considered during the *Evolving Strategy* phase (Tables 2 and Appendix B.6).

Across most models, decision-making capability does not scale proportionally with the size of the environment. Instead, performance remains relatively flat despite a substantial increase in the number of available options. Notably, Gemini-3 (Fast), which supports the largest maximum context window, performs better than other models in this respect, suggesting that larger context capacity helps retain salient information even when the effective interaction context is constrained.

Nevertheless, even the strongest models fail to cover the full decision space. This indicates that current systems are unable to expand their effective decision-making scope as the environment grows, leading to systematic performance degradation in larger and more complex settings.

6.2 Incomplete Decision-Making Due to Limited Information Coverage

We analyze the information sources that models attend to when making operational decisions by examining the data queried for the set of SKUs included in each day's final strategy. This analysis reveals a clear concentration of attention on a limited subset of signals.

Model	Supplier	Inventory	Return	Rating	Price	Review	Sales	History
DeepSeek-V3.2 (Exp.)	41.4	100.0	26.7	75.4	6.6	0.6	89.2	0.0
Gemini-3 (Fast)	69.6	100.0	41.0	82.3	67.7	6.3	87.6	2.9
GLM-4.6	93.9	98.3	79.8	77.8	84.1	5.1	96.3	0.0
OpenAI-5.1 Mini	58.8	99.0	5.1	78.6	3.9	10.6	84.7	0.3
Grok-4.1 Fast	89.9	99.8	84.0	94.0	88.6	36.0	96.4	0.3
Kimi-K2 (Thinking)	57.2	90.4	25.9	51.4	36.2	11.0	64.0	0.5
Qwen-235B (Thinking)	76.8	78.3	31.5	58.2	19.8	23.6	74.6	0.0
Average	69.7	95.1	42.0	74.0	43.8	13.3	84.7	1.0

Table 3: Percentage of days on which each model queries specific information sources when making decisions for SKUs included in the final daily strategy. Higher values indicate greater reliance on the corresponding data source.

	Macro Strategy			Execution Strategy		
Model	Std_diff (↓)	MAC (↓)	TV (↓)	Std_diff (↓)	MAC (↓)	TV (↓)
DeepSeek-V3.2(Exp.)	0.130	0.090	5.00	0.289	0.144	7.93
Gemini-3 (Fast)	0.136	0.085	3.82	0.293	0.225	10.18
GLM-4.6	0.131	0.096	4.52	0.264	0.209	9.87
OpenAI-5.1 Mini	0.069	0.051	2.50	0.229	0.166	8.00
Grok-4.1 Fast	0.079	0.045	2.71	0.204	0.151	9.39
Kimi K2(Thinking)	0.179	0.133	**6.95**	0.335	0.271	**14.08**
Qwen-235B (Thinking)	**0.240**	**0.189**	6.51	**0.391**	**0.306**	10.57

Table 4: Temporal instability metrics of macro- and execution-level strategy similarity in the Easy environment. All metrics are lower-is-better (↓). Larger values indicate greater temporal instability. Column-wise maximum values are highlighted in bold.

Specifically, most models primarily rely on supplier prices, inventory levels, SKU ratings, and historical sales records when deciding how to operate selected SKUs in Table 3. In contrast, several other critical signals—such as recent customer reviews, return rates, and current selling prices—are consistently underutilized or entirely ignored.

Further correlation analysis shows a strong positive relationship between the frequency of SKU reviews queries and average daily sales performance in Appendix B.5. This observation aligns with the underlying environment dynamics and indicates that incomplete information coverage is a key factor limiting decision quality. Overall, these results suggest that models often fail to perform sufficiently comprehensive information gathering, leading to systematically suboptimal operational decisions.

6.3 Temporal Instability in Execution-Level Decision-Making

Beyond limitations in decision scalability and information coverage, we identify temporal instability in execution-level decision-making as a key contributor to long-horizon failure. Even under relatively stable environmental conditions, agents frequently revise their strategies across consecutive days, leading to inconsistent execution trajectories.

Measuring Strategy Similarity. We quantify temporal instability by measuring the similarity between strategies on adjacent days at both macro and execution levels. Macro strategy similarity is assessed using an LLM-based prompt that evaluates semantic consistency between consecutive high-level plans. Execution strategy similarity is computed via set-based Jaccard similarity over key fields, including `focus_skus`, `sku_supplier_mapping`, `news_to_monitor`, and `sku_to_monitor`, with the final execution similarity obtained by averaging across fields.

Instability Metrics. To characterize temporal fluctuations, we compute three complementary metrics: the standard deviation of first-order differences (*Std_diff*) to capture short-term volatility, the mean absolute change (*MAC*) to estimate typical day-to-day variation, and total variation (*TV*) to reflect cumulative long-term instability.

Across all three environment configurations, we observe consistent temporal instability in both macro- and execution-level strategies. For clarity, we present detailed results from the Easy environment in Table 4 and Appendix B.4 . Despite its reduced complexity, the Easy setting already exhibits pronounced temporal fluctuations. In particular, macro strategies remain relatively stable over time, whereas execution strategies show substantially larger variability across most models. This effect is especially pronounced for Qwen-235B (Thinking), which also performs poorly across all environments.

Overall, these results indicate that long-horizon failures arise not only from suboptimal strategy formulation, but more fundamentally from the inability to maintain temporally consistent execution policies over extended horizons.

6.4 Hallucinations and Invalid Actions

Finally, manual inspection reveals recurrent failure patterns that directly break planning correctness and action validity. We distinguish two closely related but practically different issues:

Hallucinations in reasoning Models occasionally generate reasoning traces that reference nonexistent SKUs or fabricate numerical quantities, and subsequently incorporate these hallucinated elements into multi-step plans (examples are provided in Appendix B.1). As a result, decisions become misaligned with the true environment state, even when the overall planning structure appears internally coherent.

Invalid or irrational actions. Models sometimes output actions that violate basic constraints or are inconsistent with historical demand, such as negative order quantities or implausible pricing in Appendix B.2. Even though state-modifying operations are restricted to only a small set of tools, these invalid actions occur with non-negligible frequency and can quickly destabilize the system in long-horizon operation.

In realistic operational settings, both hallucinations and invalid actions would be unacceptable and could directly trigger system collapse.

7 Related Work

Retail and supply-chain decision benchmarks. Recent work on retail and supply-chain decision-making has evolved from isolated subproblems toward integrated, multi-stage environments. Reinforcement-learning benchmarks such as OFCOURSE, GymSC-style simulators, and MARLIM (Zhu et al., 2023; Shar et al., 2022; Leluc et al., 2023) model fulfillment and inventory control under demand uncertainty, demonstrating the effectiveness of learned policies in mitigating effects such as demand volatility. However, these environments typically emphasize fixed structures and short- to medium-horizon policies, limiting their ability to evaluate sustained strategic behavior. More recent benchmarks incorporating LLM-based agents, including InvAgent and AIM-Bench (Quan and Liu, 2025; Zhao et al., 2025), begin to explore language-driven reasoning and decision biases, yet still fall short in assessing long-horizon, strategy-aware autonomy in realistic retail settings.

Long-horizon planning benchmarks for LLMs. In parallel, a growing body of benchmarks targets long-horizon planning abilities of large language models across structured and interactive environments. PlanBench focuses on classical plan generation, while WebShop, Mind2Web, and ScienceWorld (Deng et al., 2023; Wang et al., 2022; Yao et al., 2023) evaluate multi-step interaction, error recovery, and adaptive behavior in dynamic settings. More recent benchmarks—such as HeroBench, OdysseyBench, UltraHorizon (Luo et al., 2025; Anokhin et al., 2025; Wang et al., 2025), and explicitly stress extended, interdependent decision sequences that require persistent memory, hierarchical reasoning, and long-term strategy maintenance. Collectively, these benchmarks suggest that while short-horizon tasks are increasingly tractable, robust long-horizon planning and execution remain a central challenge for LLM-based agents.

Long-horizon agent frameworks. Recent advances in agent design address these challenges by introducing structured frameworks that decouple high-level planning from low-level execution. Approaches such as Plan-and-Act (Erdogan et al., 2025) and EAGLET (Si et al., 2025) adopt planner–executor architectures to support hierarchical decision-making, dynamic replanning, and improved execution stability over long horizons. Extensions to multi-agent settings ELHPlan (Ling et al., 2025) and plan-aware context management frameworks PAACE (Yuksel, 2025) further emphasize task decomposition, explicit strategy representation, and memory-aware context control. Together, these works indicate that scalable long-horizon autonomy increasingly relies on structured planning modules and controlled execution mechanisms rather than monolithic end-to-end policies.

8 Conclusion

This paper introduces RetailBench, a high-fidelity benchmark for evaluating long-horizon autonomous decision-making in realistic retail environments. RetailBench models supermarket operations as a stochastic, multi-factor, and temporally extended process, requiring agents to jointly reason about pricing, inventory, information acquisition, and financial sustainability over extended horizons. We further propose the Evolving Strategy & Execution framework, which decouples high-level strategy evolution from low-level execution to better support long-horizon autonomy.

Experiments across seven state-of-the-art large language models show that the proposed framework consistently improves operational stability and economic performance compared to a Reflection-based baseline. Nevertheless, performance degrades sharply as environment complexity increases, exposing persistent limitations in decision scalability, information utilization, execution stability, and action validity. These findings indicate that while structured agent frameworks alleviate some challenges, current LLM-based agents remain far from robust, strategy-aware autonomy in complex dynamic environments. RetailBench provides a principled testbed for advancing research on long-horizon decision-making and agentic reasoning.

9 Limitations

Despite its realism and scale, this work has several limitations. First, RetailBench focuses on a single-store supermarket setting; while expressive, it does not capture multi-store coordination, competitive markets, or strategic interactions among multiple autonomous agents. Second, although the environment incorporates stochastic demand, news dynamics, and supply-chain delays, it remains a simulation grounded in historical data and simplified economic assumptions, which may not fully reflect the complexities of real-world retail systems.

Third, our evaluation is limited to prompting-based LLM agents without parameter updates or long-term learning across episodes; stronger performance may be achievable through reinforcement learning, fine-tuning, or hybrid neuro-symbolic approaches. Finally, while we identify key failure modes such as hallucinations and economically irrational actions, we do not propose explicit algorithmic mechanisms to enforce economic constraints or factual grounding during execution. Addressing these limitations—through richer environments, multi-agent extensions, learning-based adaptation, and constraint-aware action control—remains an important direction for future research.

References

Dario Amodei. 2024. Machines of loving grace. https://www.darioamodei.com/essay/machines-of-loving-grace. Accessed: 2025-12-01.

Andon Labs. 2025. Vending-bench 2: A benchmark for long-horizon business simulation. https://andonlabs.com/evals/vending-bench-2. Accessed: 2025-12-10.

Petr Anokhin, Roman Khalikov, Stefan Rebrikov, Viktor Volkov, Artyom Sorokin, and Vincent Bissonnette. 2025. Herobench: A benchmark for long-horizon planning and structured reasoning in virtual worlds. *Preprint*, arXiv:2508.12782.

ashraq. 2025. financial-news-articles. https://huggingface.co/datasets/ashraq/financial-news-articles. Accessed: 2025-12-01.

Axel Backlund and Lukas Petersson. 2025. Vending-bench: A benchmark for long-term coherence of autonomous agents. *Preprint*, arXiv:2502.15840.

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, and 245 others. 2025. Deepseek-v3.2: Pushing the frontier of open large language models. *Preprint*, arXiv:2512.02556.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. *Preprint*, arXiv:2306.06070.

Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-act: Improving planning of agents for long-horizon tasks. *Preprint*, arXiv:2503.09572.

Dave Fedewa, Chris Holder, Wynn Teichner, and Ben Wiseman. 2021. Five-star growth: Using online ratings to design better products. *McKinsey & Company*. Accessed: 2025-11-30.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *Preprint*, arXiv:2410.07985.

Google DeepMind. 2024. Gemini 3 flash (preview). https://deepmind.google/technologies/gemini/. Large multimodal language model, preview version.

Dhruv Grewal, Jens Nordfält, Anne Roggeveen, Rainer Olbrich, and Hans Christian Jansen. 2014. Price-quality relationship in pricing strategies for private labels. *Journal of Product and Brand Management*, 23(6):429–438.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. Swe-bench: Can language models resolve real-world github issues? *Preprint*, arXiv:2310.06770.

University of Chicago Booth School of Business Kilts Center for Marketing. Dominick's dataset. https://www.chicagobooth.edu/research/kilts/research-data/dominicks. Accessed: 2025-11-01.

Thomas Kwa, Ben West, Joel Becker, Amy Deng, Katharyn Garcia, Max Hasin, Sami Jawhar, Megan Kinniment, Nate Rush, Sydney Von Arx, Ryan Bloom, Thomas Broadley, Haoxing Du, Brian Goodrich, Nikola Jurkovic, Luke Harold Miles, Seraphina Nix, Tao Lin, Neev Parikh, and 6 others. 2025. Measuring ai ability to complete long tasks. *Preprint*, arXiv:2503.14499.

Rémi Leluc, Elie Kadoche, Antoine Bertoncello, and Sébastien Gourvénec. 2023. Marlim: Multi-agent

reinforcement learning for inventory management. *Preprint*, arXiv:2308.01649.

Shaobin Ling, Yun Wang, Chenyou Fan, Tin Lun Lam, and Junjie Hu. 2025. Elhplan: Efficient long-horizon task planning for multi-agent collaboration. *Preprint*, arXiv:2509.24230.

Haotian Luo, Huaisong Zhang, Xuelin Zhang, Haoyu Wang, Zeyu Qin, Wenjie Lu, Guozheng Ma, Haiying He, Yingsha Xie, Qiyang Zhou, Zixuan Hu, Hongze Mi, Yibo Wang, Naiqiang Tan, Hong Chen, Yi R. Fung, Chun Yuan, and Li Shen. 2025. Ultrahorizon: Benchmarking agent capabilities in ultra long-horizon scenarios. *Preprint*, arXiv:2509.21766.

Daniel McFadden. 1974. Conditional logit analysis of qualitative choice behavior. In Paul Zarembka, editor, *Fontiers in Econometrics*, pages 105–142. Academic press, New York.

METR. 2025. Measuring ai ability to complete long tasks. METR blog.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general ai assistants. *Preprint*, arXiv:2311.12983.

Nof1.ai. 2025. Alpha arena — exploring the limits of large language models as quant traders. `https://nof1.ai/blog/TechPost1`. Accessed: 2025-12-10.

OpenAI. 2025. Gpt-5 mini. `https://platform.openai.com/docs/models/gpt-5-mini`. Large language model — cost-efficient GPT-5 variant.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, Michael Choi, Anish Agrawal, Arnav Chopra, Adam Khoja, Ryan Kim, Richard Ren, Jason Hausenloy, Oliver Zhang, Mantas Mazeika, and 1093 others. 2025. Humanity's last exam. *Preprint*, arXiv:2501.14249.

Yinzhu Quan and Zefang Liu. 2025. Invagent: A large language model based multi-agent system for inventory management in supply chains. *Preprint*, arXiv:2407.11384.

Ibrahim Shar, Wenhuan Sun, Haiyan Wang, and Chetan Gupta. 2022. Deep reinforcement learning toward robust multi-echelon supply chain inventory optimization. pages 1385–1391.

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

Shuzheng Si, Haozhe Zhao, Kangyang Luo, Gang Chen, Fanchao Qi, Minjia Zhang, Baobao Chang, and Maosong Sun. 2025. A goal without a plan is just a wish: Efficient and effective global planner training for long-horizon agent tasks. *Preprint*, arXiv:2510.05608.

GLM Team, Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, Kedong Wang, Lucen Zhong, Mingdao Liu, Rui Lu, Shulin Cao, Xiaohan Zhang, Xuancheng Huang, Yao Wei, and 152 others. 2025a. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *Preprint*, arXiv:2508.06471.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, Chenzhuang Du, Dikang Du, Yulun Du, Yu Fan, and 150 others. 2025b. Kimi k2: Open agentic intelligence. *Preprint*, arXiv:2507.20534.

Qwen Team. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

The Terminal-Bench Team. 2025b. Terminal-bench: A benchmark for ai agents in terminal environments.

Mark D. Uncles. 1987. Discrete choice analysis: Theory and application to travel demand. *Journal of the Operational Research Society*, 38(4):370–371.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. Scienceworld: Is your agent smarter than a 5th grader? *Preprint*, arXiv:2203.07540.

Weixuan Wang, Dongge Han, Daniel Madrigal Diaz, Jin Xu, Victor Rühle, and Saravan Rajmohan. 2025. Odysseybench: Evaluating llm agents on long-horizon complex office application workflows. *Preprint*, arXiv:2508.09124.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. Browsecomp: A simple yet challenging benchmark for browsing agents. *Preprint*, arXiv:2504.12516.

xAI. 2025. Grok 4.1 fast and agent tools api. `https://x.ai/news/grok-4-1-fast`. Accessed [Your Access Date].

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2023. Webshop: Towards scalable real-world web interaction with grounded language agents. *Preprint*, arXiv:2207.01206.

Kamer Ali Yuksel. 2025. Paace: A plan-aware automated agent context engineering framework. *Preprint*, arXiv:2512.16970.

Xuhua Zhao, Yuxuan Xie, Caihua Chen, and Yuxiang Sun. 2025. Aim-bench: Evaluating decision-making biases of agentic llm as inventory manager. *Preprint*, arXiv:2508.11416.

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. *Preprint*, arXiv:2307.13854.

Yiheng Zhu, Yang Zhan, Xuankun Huang, Yuwei Chen, yujie Chen, Jiangwen Wei, Wei Feng, Yinzhi Zhou, Haoyuan Hu, and Jieping Ye. 2023. Ofcourse: A multi-agent reinforcement learning environment for order fulfillment. In *Advances in Neural Information Processing Systems*, volume 36, pages 34765–34777. Curran Associates, Inc.

A Environment Configuration Details

A.1 State Space Decomposition

We construct the environment using real-world retail data from the Dominick's dataset (Kilts Center for Marketing). From the 20 available product categories, we select 96 SKUs with the most complete and informative sales records. The richness of these observations enables reliable modeling of the relationship between pricing decisions and realized demand.

Key Symbols. Let $\mathcal{J}$ denote the set of selected SKUs, with cardinality $|\mathcal{J}| = J$ (where $J = 96$ in our experiments). Each SKU $j \in \mathcal{J}$ belongs to a product category $\mathrm{cat}(j) \in \{1, \ldots, 20\}$. For each SKU j, let $\mathcal{K}(j)$ denote its associated supplier set, with $|\mathcal{K}(j)| = 5$.

A.1.1 Product State S_t^{prod}

For each SKU j, we maintain a set of static attributes together with time-varying operational signals:

$$S_{t,j}^{\text{prod}} = \big(\texttt{id}_j,\ \texttt{desc}_j,\ L_j,\ p_{jt},\ \mathbf{h}_{j,t}^{\text{sales}},\ \mathbf{r}_{j,t}\big). \quad (1)$$

Here, $\texttt{id}_j$ and $\texttt{desc}_j$ denote the unique identifier and textual description of SKU j, respectively. L_j denotes the shelf life (in days), and p_{jt} is the retail price on day t. The term $\mathbf{h}_{j,t}^{\text{sales}}$ encodes historical sales statistics, while $\mathbf{r}_{j,t}$ summarizes aggregated customer review signals, such as average rating and review volume.

Data grounding. The SKU set and cost priors are constructed from the Dominick's dataset (Kilts Center for Marketing). Historical sales records from the same source are used to calibrate the demand model.

A.1.2 Inventory State S_t^{inv}

Inventory is represented at the SKU level with age tracking. Let I_{jt} denote the on-hand units of SKU j at the beginning of day t. To support shelf-life constraints and depreciation, we track the arrival time and remaining shelf life of each unit.

Capacity constraint and pending queue. Let Cap denote the total inventory capacity of the store. If incoming replenishment orders would violate this constraint, excess units are placed into a first-in-first-out (FIFO) pending queue $\mathcal{Q}_t$ and become available only after sufficient inventory space is released:

$$\sum_{j \in \mathcal{J}} \sum_{a=0}^{L_j} I_{jt}^{(a)} \le \mathrm{Cap}. \quad (2)$$

Stockout signal. When realized demand exceeds the available sellable inventory of SKU j on day t, a stockout indicator is triggered. This signal is recorded at the end of day t and exposed to the agent as explicit operational feedback.

Expiration and destruction. At each day transition, inventory units whose residence time exceeds their shelf life are removed from the system.

A.1.3 Supply Chain State S_t^{sup}

Each SKU j is associated with a set of suppliers $k \in \mathcal{K}(j)$. The supplier state includes procurement price c_{jk}, quality level q_{jk}, and lead-time distribution:

$$S_{t,j}^{\text{sup}} = \big\{(c_{jk}, q_{jk}, \mathcal{L}_{jk}) : k \in \mathcal{K}(j)\big\}. \quad (3)$$

Lead time $\ell_{jk} \sim \mathcal{L}_{jk}$ is sampled when an order is placed. Procurement prices are discretized into tiers using Dominick's cost signals (Kilts Center for Marketing), following empirically observed positive correlations between price tier and quality level (Grewal et al., 2014).

Enforced diversity. To avoid degenerate supplier configurations, we enforce that each SKU has (i) one supplier with maximal quality, (ii) one supplier with minimal price and minimal quality, and remaining suppliers spanning intermediate price–quality levels.

A.1.4 Demand Signals and Demand Generation

Demand-related components capture both observable market signals and the stochastic process governing realized demand. We define

$$S_t^{\text{dem}} = \big(N_t,\ \{\mathbf{r}_{j,t}\}_{j \in \mathcal{J}}\big), \quad (4)$$

where N_t denotes daily customer traffic.

Customer traffic. Daily customer traffic is derived from the Dominick's dataset.

Consumer choice and demand generation. Given customer traffic N_t, prices $\{p_{jt}\}_{j\in\mathcal{J}}$, review signals $\{\mathbf{r}_{j,t}\}$, and external news $\mathcal{E}_t$, we generate realized demand using a discrete-choice model. Following standard practice in empirical economics, we adopt a Multinomial Logit (MNL) framework (McFadden, 1974; Uncles, 1987).

For SKU j, the raw utility is defined as

$$U_{jt}^{\text{raw}} = \alpha_j + \beta_j p_{jt} + \varepsilon_{jt}, \tag{5}$$

where α_j captures intrinsic preference, β_j models price sensitivity, and ε_{jt} represents idiosyncratic shocks.

We further incorporate review effects, news signals, and within-category substitution:

$$\tilde{U}_{jt} = U_{jt}^{\text{raw}} + \Delta_j(\mathbf{r}_{j,t}, \mathcal{E}_t) + \sum_{\substack{i\neq j \\ \text{cat}(i)=\text{cat}(j)}} \gamma_{ji} \exp(\tilde{U}_{it}). \tag{6}$$

With the outside option utility normalized to zero, the purchase probability for SKU j on day t is

$$p_{jt}^* = \frac{\exp(\tilde{U}_{jt})}{1 + \sum_{i=1}^{J} \exp(\tilde{U}_{it})}. \tag{7}$$

Realized demand. Given traffic N_t and purchase probabilities $\{p_{jt}^*\}$, potential demand is sampled as

$$y_{jt} \sim \text{Binomial}(N_t,\ p_{jt}^*), \tag{8}$$

and realized sales are capped by available on-hand inventory.

A.1.5 External Information S_t^{ext} (News Module)

We maintain a set of active news events $\mathcal{E}_t$. Each event $e \in \mathcal{E}_t$ is represented as

$$e = (\texttt{type},\ \texttt{scope},\ \texttt{target},\ \texttt{side},\ \texttt{sign},\ \eta,\ \texttt{text},\ \texttt{ttl}) \tag{9}$$

where $\texttt{scope} \in \{\text{macro}, \text{category}, \text{product}, \text{neutral}\}$, $\texttt{side} \in \{\text{demand}, \text{supply}, \text{both}\}$, $\texttt{sign} \in \{+1, -1\}$, η denotes impact magnitude, and $\texttt{ttl}$ is the time-to-live in days. News texts are synthesized using an LLM to resemble financial news corpora (ashraq, 2025).

Impact application. News impacts are incorporated into (i) demand utilities and/or (ii) supplier prices depending on `side` and `scope`. Neutral news has $\eta = 0$ by design.

A.1.6 Financial State S_t^{fin}

Let F_t denote available funds at the start of day t. Net worth is computed as

$$\text{NW}_t = F_t + \sum_{j\in\mathcal{J}} \sum_{a=0}^{L_j} I_{jt}^{(a)} \cdot v_j(a), \tag{10}$$

where $v_j(a)$ denotes the per-unit value at age a. We adopt linear shelf-life depreciation:

$$v_j(a) = c_j \cdot \max\left(0,\ 1 - \frac{a}{L_j}\right), \tag{11}$$

with c_j denoting a reference procurement cost (e.g., the mean or realized supplier cost).

A.2 Policy Detail

A.2.1 Policy Presentation

To support structured and interpretable decision making, we represent the agent policy using a hierarchical abstraction that separates strategic intent from executable actions. Specifically, each policy is composed of three layers:

1. **Macro Strategy**, which captures high-level managerial principles that persist across days;
2. **Execution Strategy**, which encodes structured operational guidance in a machine-readable form;
3. **Daily Actions**, which enumerate concrete executable operations submitted to the environment.

This design enables the agent to reason at different temporal and semantic granularities, while maintaining a clear separation between planning and execution.

Macro Strategy. The macro strategy consists of a set of natural-language statements that describe high-level objectives (e.g., prioritizing inventory turnover or focusing on high-margin products). These statements are non-executable and serve as persistent guidance for downstream decision making.

Execution Strategy. The execution strategy consists of six key components. `focus_skus` specifies the SKUs that require immediate attention. `sku_supplier_mapping` denotes the corresponding suppliers for each SKU. `news_to_monitor` identifies relevant news signals to track. `skus_to_reorder` indicates SKUs that require replenishment. `price_adjustment` specifies the SKUs whose prices should be adjusted along with the corresponding adjustment magnitudes. `sku_to_monitor` denotes SKUs under observation. The `other` field captures additional execution directives.

Daily Actions. Daily actions consist of two operation types: `place_order`, which places purchase orders for selected SKUs, and `modify_sku_price`, which adjusts the prices of specified SKUs.

Strategy–Execution Protocol. Policy usage follows a two-phase protocol. In the *strategy phase*, the agent analyzes the environment and constructs its macro and execution strategies. In the subsequent *execution phase*, the finalized strategy is consumed to generate executable daily actions. The strategy is immutable during execution, enforcing a clear separation between deliberation and action.

A.2.2 Policy Example

See in Table 5.

A.3 Environment Setting

A.3.1 Easy Environment Configuration

The `Easy` configuration is designed for simpler scenarios with limited resources and a reduced category set.

- **Time Range:**
  - Data begin time: 06/06/91
  - Data end time: 12/31/95
  - Store begin time: 09/07/91
  - Store ID: 15
- **Financial Parameters:**
  - Initial funds: 10,000
  - Daily rent: 250
  - Inventory capacity: 10,000
- **Feature Enablement:**
  - Review enabled: `True`
  - Review ratio: 0.02
  - News enabled: `False`
- **Selected Categories (5 categories):**
  - Bathroom_Tissues
  - Canned_Soup
  - Cigarettes
  - Front_end_candies
  - Soft_Drinks
- **Category Effects:** All categories have a uniform effect of -0.2
- **Random Seed:** 42 (for reproducibility)

A.3.2 Middle Environment Configuration

The `Middle` configuration uses static data but with full resource capacity and all categories enabled.

- **Time Range:** Same as `Easy`
- **Financial Parameters:**
  - Initial funds: 50,000
  - Daily rent: 1,000
  - Inventory capacity: 40,000
- **Feature Enablement:**
  - Review enabled: `True`
  - Review ratio: 0.02
  - News enabled: `False`
- **Selected Categories (20 categories):**
  - Bathroom_Tissues, Beer, Bottled_Juices, Canned_Soup, Canned_Tuna
  - Cereals, Cheeses, Cigarettes, Cookies, Crackers
  - Dish_Detergent, Fabric_Softeners, Front_end_candies, Frozen_Entrees
  - Frozen_Juices, Oatmeal, Paper_Towels, Snack_Crackers
  - Soft_Drinks, Toothpastes
- **Category Effects:** All categories have a uniform effect of -0.2
- **Random Seed:** 42

A.3.3 Hard Environment Configuration

The `Hard` configuration uses dynamic data with news events enabled, providing the most complex simulation environment.

- **Time Range:** Same as other configurations
- **Financial Parameters:**

- Initial funds: 50,000
- Daily rent: 1,000
- Inventory capacity: 40,000

- **Feature Enablement:**
  - Review enabled: `True`
  - Review ratio: 0.02
  - News enabled: `True`
- **News Configuration:**
  - News impact base scale: 0.4
  - News daily count: 20
  - News random seed: 42
  - News sample ratios:
    - Neutral: 0.9
    - Single category: 0.02
    - Macro all: 0.03
    - SKU level: 0.05
  - News impact mode weights:
    - Neutral: 0.0
    - Macro all: 1.0
    - Single category: 1.0
    - SKU level: 1.2
- **Selected Categories:** Same 20 categories as `Middle`
- **Category Effects:** All categories have a uniform effect of -0.2
- **Random Seed:** 42

A.4 Environment Simulation

See in Figure 3

B Rollout Details

B.1 Hallucinations in Reasoning and Planning

This appendix provides representative examples of hallucinations observed during long-horizon rollouts, focusing on reasoning-time errors that propagate into multi-step planning despite internally coherent logic.

B.1.1 Non-existent SKUs

During execution, models occasionally reference or plan around SKUs that do not exist in the environment. Across all rollouts, we identify 14 distinct non-existent SKUs that repeatedly appear in final daily strategies, indicating persistent hallucinations rather than isolated parsing or formatting errors. Representative examples include SKU identifiers such as `10700013100`, `166051312`, and `440004627`.

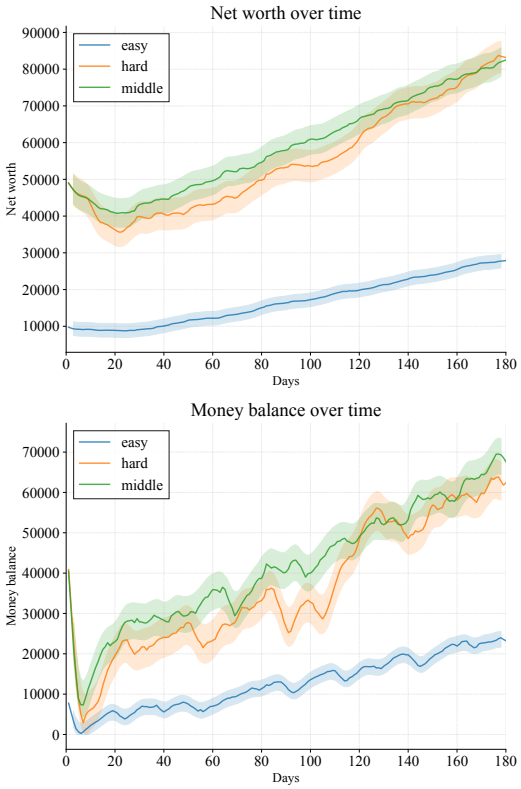

Figure 3: Net worth and available funds trajectories of the heuristic policy under different environment configurations, illustrating the calibrated difficulty levels. The *Middle* and *Hard* settings involve a larger number of product categories, enabling higher potential net worth and cash accumulation over the course of an episode.

Example. In the following strategy output, the model includes a non-existent SKU (`440004627`) in its set of focus SKUs. This hallucinated identifier is subsequently treated as a valid product in downstream reasoning and planning:

```
{
  "day": 14,
  "current_date": "1991-09-22",
  "strategy": {
    "macro_strategy": ***,
    "execute_strategy": {
      "focus_skus": [
        "3000001460",
        "3700063037",
        "440004627",
        "5100001251"
      ],
      "news_to_monitor" ***,
    },
    "today_action": ***,
  }
}
```

B.1.2 Hallucinated Dates

Models also hallucinate temporal information during tool-based reasoning when required to provide calendar dates that are not explicitly specified by the environment. Instead of querying the current simulation date, the model resolves this underspecification by fabricating a plausible calendar mapping in order to proceed with execution.

Example. The following excerpt illustrates the model's reasoning process when attempting to issue a tool call that requires valid date strings:

```
Tool requires dates in YYYY-MM-DD
or MM/DD/YY.
Only day index is known: Day 14.
No calendar start date is provided.

Assume Day 1 = 2023-01-01.
Infer Day 14 = 2023-01-14.

start_date = 2023-01-13
end_date   = 2023-01-14
```

Based on this fabricated assumption, the model issues a syntactically valid tool call using the inferred dates. Although the reasoning chain is internally consistent, the assumed calendar mapping is not grounded in the true environment state, resulting in a semantically incorrect interaction.

These examples highlight a recurring failure mode in which models resolve underspecified variables through plausible fabrication rather than explicit information acquisition, leading to planning trajectories that appear coherent yet diverge from the actual environment dynamics.

B.2 Invalid or Irrational Actions

We identify invalid or economically irrational actions by applying a set of heuristic rules to model-issued tool calls. Specifically, we flag actions that violate basic economic or operational constraints, including: (i) setting product prices to zero, negative values, or unrealistically high levels (e.g., exceeding 50); and (ii) placing orders with quantities far beyond plausible operational scales for a single SKU.

Across all evaluated rollouts, we detect more than 300 such anomalous tool calls. Among them, 197 correspond to invalid or irrational price modifications, while 125 involve excessively large ordering decisions. These behaviors are observed across multiple models and environment configurations.

Excessive Ordering. The following example is taken from a rollout of the *Kimi K2 Thinking* model in the *Hard* environment. The model places an implausibly large order for a single SKU, far exceeding realistic replenishment volumes:

```
tool: place_order
model: Kimi K2 Thinking
sku_id: 5100000011
quantity: 18000
```

Although syntactically valid, such actions introduce extreme inventory shocks that are misaligned with realistic retail operations.

Invalid Price Modifications. Irrational pricing behavior is also observed across models. In the following example, the model updates the price of a SKU by several orders of magnitude (log excerpted from `still_hard/kimi_thinking/tool_calls.jsonl`):

```
tool: modify_sku_price
model: Kimi K2 Thinking
old_price: 0.25
new_price: 999.0
```

In other cases, models attempt to assign zero or negative prices, which are explicitly rejected by the environment. While such invalid actions are prevented at execution time, extreme but valid values can still propagate downstream and destabilize subsequent decision-making.

Overall, these results indicate that current LLM-based agents lack reliable internal mechanisms for enforcing basic economic plausibility at the action level, even when tool interfaces enforce syntactic validity and detailed execution logs are available for inspection.

B.3 Rollout Results

See in Table 6

B.4 Strategy Score Analysis

See in Figure 4, 5

B.5 Tool use Analysis

See in Figure 6

B.6 Category Analysis

See in Table 7

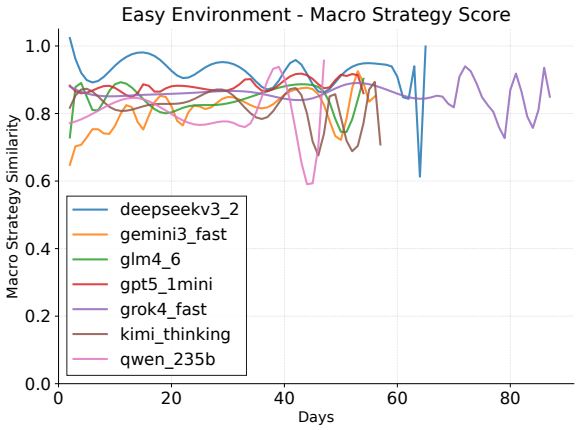

Figure 4: Macro strategy similarity over time in the easy environment. Higher values indicate greater consistency in high-level decisions across days.

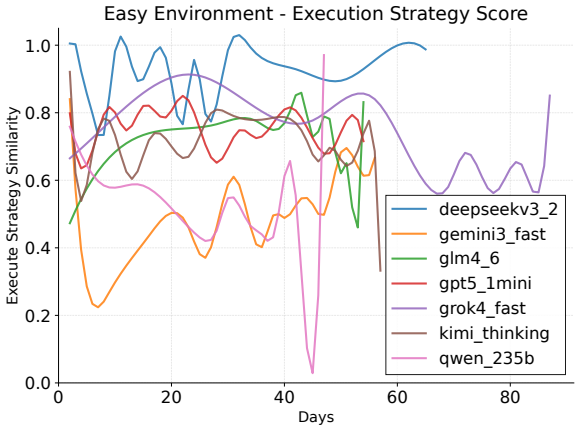

Figure 5: Execution strategy similarity over time in the easy environment. Execution-level behaviors exhibit substantially higher temporal variability than macro strategies.

C Prompt

C.1 Evolving Strategy & Execution Framework Evolving Strategy Phase System Prompts

```
You are a retail strategy analyst. Your
↪ task is to analyze current business
↪ data and determine whether the
↪ current strategy needs adjustment.


Environment Characteristics

- The store operates with a large number
↪ of SKUs, where products within the
↪ same category interact and may
↪ substitute or cannibalize each
↪ other's demand.
```

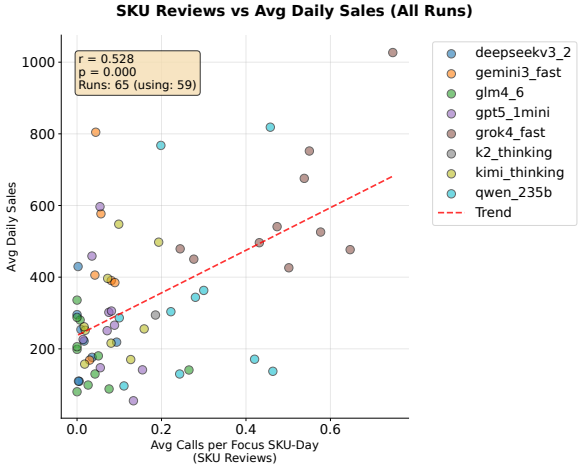

Figure 6: Correlation between the frequency of SKU review queries during rollouts and average daily sales across all runs. Each point represents a single rollout. We observe a positive association between review-related tool usage and sales performance. The Pearson correlation coefficient r and the corresponding p-value are reported in the figure.

```
- Historical sales data provides
↪ essential signals for future
↪ decision-making.
- Customer reviews dynamically influence
↪ product demand and sales velocity,
↪ with recent reviews having stronger
↪ effects.
{news_characteristic}
- Supply chains involve delivery lead
↪ times, requiring forward-looking
↪ inventory planning.
- Orders require delivery time: When you
↪ place an order (place_order), the
↪ items will not arrive immediately.
↪ The delivery time varies and can
↪ take up to 7 days (within 7 days).
↪ You should plan your inventory
↪ accordingly and account for this
↪ lead time when making ordering
↪ decisions. Orders placed today will
↪ arrive within 7 days, but the exact
↪ arrival time is variable.
- Inventory items depreciate in value
↪ over time and may require disposal
↪ when approaching expiration.
- Supplier heterogeneity affects product
↪ quality perceptions and customer
↪ reviews, leading to
↪ supplier-dependent demand outcomes.
```

```
- Daily rent: The store incurs a fixed
↪  daily rent cost of {daily_rent} that
↪  must be paid each day. This daily
↪  operating cost is automatically
↪  deducted at the end of each day and
↪  makes cash-flow management critical
↪  for long-term survival and
↪  profitability. You must ensure
↪  sufficient funds are available to
↪  cover the daily rent. The daily rent
↪  amount is fixed and must be paid
↪  every single day, regardless of
↪  sales performance or other factors.

# Your Role in Strategy Phase

Each day starts with a STRATEGY PHASE
↪  where you:
1. Review the current strategy (provided
↪  at the start of this phase)
2. Use data analysis tools to gather
↪  information about:
   - Current inventory status
   - Recent sales history (last 30-60
   ↪  days)
   - Customer reviews and ratings
   - Supplier prices and quality
{news_data_point}
   - Current financial status
3. Compare current situation with
↪  previous days to identify
↪  significant changes
4. Set the strategy using the three
↪  separate tools (set_macro_strategy,
↪  set_execute_strategy, set_action) to
↪  set the three strategy components

# Strategy Format

The strategy consists of three
↪  components:

1. **macro_strategy**: A list of broad
↪  strategic guidelines (array of
↪  strings)
   - Examples: ["Focus on high-margin
   ↪  products", "Maintain competitive
   ↪  pricing", "Prioritize inventory
   ↪  turnover"]

2. **execute_strategy**: An object with
↪  seven fields, all values are arrays:
   - **focus_skus**: Array of SKU IDs
   ↪  that need attention (e.g.,
   ↪  ["SKU_001", "SKU_002"])
   - **sku_supplier_mapping**: Array of
   ↪  mapping objects (e.g.,
   ↪  [{{"sku_id": "SKU_001",
   ↪  "supplier_id": "supplier_A"}},
   ↪  {{"sku_id": "SKU_002",
   ↪  "supplier_id": "supplier_B"}}])
{news_to_monitor_field}
   - **skus_to_reorder**: Array of SKU
   ↪  IDs that need reordering (e.g.,
   ↪  ["SKU_003", "SKU_004"])
   - **price_adjustments**: Array of
   ↪  price adjustment objects (e.g.,
   ↪  [{{"sku_id": "SKU_001",
   ↪  "adjustment": "increase by
   ↪  10%"}}, {{"sku_id": "SKU_002",
   ↪  "adjustment": "decrease by
   ↪  5%"}}])
   - **sku_to_monitor**: Array of SKU
   ↪  IDs that should be closely
   ↪  monitored (e.g., ["SKU_005",
   ↪  "SKU_006"])
   - **other**: Array of other strategy
   ↪  notes or metadata (e.g.,
   ↪  [{{"comment": "..."}},
   ↪  {{"risk_level": "high"}}])

3. **today_action**: An array of action
↪  objects, each representing a
↪  concrete action using the parameter
↪  format of `place_order` or
↪  `modify_sku_price`.
   - Each action MUST be an object of
   ↪  the form:
     - {{"tool": "place_order",
     ↪  "arguments": {{<place_order
     ↪  arguments>}}}}
     - OR {{"tool": "modify_sku_price",
     ↪  "arguments":
     ↪  {{<modify_sku_price
     ↪  arguments>}}}}
   - Example:
     [
       {{"tool": "place_order",
       ↪  "arguments": {{"sku_id":
       ↪  "SKU_001", "supplier_id":
       ↪  "supplier_A", "quantity":
       ↪  100}}}},
```

```
       {{"tool": "modify_sku_price",
       ↪  "arguments": {{"sku_id":
       ↪  "SKU_002", "new_price":
       ↪  9.99}}}}
     ]

# Strategy Setting Tools

Use three separate tools to set
↪  different parts of the strategy:
- **set_macro_strategy**: Set the
↪  macro_strategy (array of strings)
  - Parameter: `macro_strategy` (array
  ↪  of strings)
  - Example: set_macro_strategy(macro_s
  ↪  trategy=["Focus on high-margin
  ↪  products", "Maintain competitive
  ↪  pricing"])

- **set_execute_strategy**: Set the
↪  execute_strategy (object with seven
↪  fields, all arrays)
  - Parameter: `execute_strategy`
  ↪  (object with fields: focus_skus,
  ↪  sku_supplier_mapping{news_to_mon
  ↪  itor_param}, skus_to_reorder,
  ↪  price_adjustments, sku_to_monitor,
  ↪  other)
  - All field values must be arrays
  - Example: set_execute_strategy(execu
  ↪  te_strategy={{"focus_skus":
  ↪  ["SKU_001"],
  ↪  "sku_supplier_mapping":
  ↪  [{{"sku_id": "SKU_001",
  ↪  "supplier_id": "supplier_A"}}],
  ↪  ...}})

- **set_action**: Set the today_action
↪  (array of action objects)
  - Parameter: `action` (array of
  ↪  objects, each with "tool" and
  ↪  "arguments" fields)
  - Each action object: {{"tool":
  ↪  "place_order" |
  ↪  "modify_sku_price", "arguments":
  ↪  {{...}}}}
  - Example:
  ↪  set_action(action=[{{"tool":
  ↪  "place_order", "arguments":
  ↪  {{"sku_id": "SKU_001",
  ↪  "supplier_id": "supplier_A",
  ↪  "quantity": 100}}}}])

You can call these tools multiple times
↪  to build or modify the strategy.
↪  After your analysis, set all three
↪  components to reflect your
↪  decisions.

# Available Tools for Analysis

The available function signatures are
↪  provided within <tools></tools> XML
↪  tags:
<tools>
{tool_definitions}
</tools>

For each function call, return a JSON
↪  object with function name and
↪  arguments inside
↪  <tool_call></tool_call> XML tags:
<tool_call>
{{"name": <function-name>, "arguments":
↪  <args-json-object>}}
</tool_call>

# Important Analysis Tools

Use these tools to gather data:
- view_funds_and_date: Check current
↪  funds and date
- view_inventory: Check current
↪  inventory levels
- view_sku_sales_history: Analyze sales
↪  trends (use last 30-60 days)
- view_sku_avg_ratings: Check customer
↪  satisfaction
- view_current_date_supplier_prices:
↪  Check supplier availability and
↪  prices
{news_tools_list}
- view_current_orders: Check pending
↪  orders
- set_macro_strategy: Set the macro
↪  strategy (array of strings)
- set_execute_strategy: Set the execute
↪  strategy (object with seven fields,
↪  all arrays)
- set_action: Set the today action
↪  (array of action objects)
```

```
Note: You CANNOT use place_order or
↪ modify_sku_price in the strategy
↪ phase. These tools are only
↪ available in the execution phase.

After completing your analysis and
↪ updating the strategy, the system
↪ will transition to the EXECUTION
↪ PHASE.
```

C.2 Evolving Strategy & Execution Framework Phase System Prompts

```
You are a retail operations agent
↪ executing daily operations based on
↪ the current strategy.

# Your Role in Execution Phase

In the EXECUTION PHASE, you will receive
↪ the **final strategy** determined in
↪ the Strategy Phase. This strategy
↪ includes:
- **macro_strategy**: Broad strategic
↪ guidelines (array of strings)
- **execute_strategy**: Specific
↪ operational details (object with
↪ seven fields, all arrays)
- **today_action**: Concrete actions to
↪ take today (array of action objects)

# Important Operational Constraints

- **Daily rent**: The store incurs a
↪ fixed daily rent cost of
↪ {daily_rent} that must be paid each
↪ day. This daily operating cost is
↪ automatically deducted at the end of
↪ each day. Ensure you have sufficient
↪ funds to cover this daily expense.
↪ The daily rent amount is fixed and
↪ must be paid every single day,
↪ regardless of sales performance or
↪ other factors. This makes cash-flow
↪ management critical for long-term
↪ survival and profitability.
- **Order delivery time**: When you
↪ place an order using place_order,
↪ the items will not arrive
↪ immediately. The delivery time
↪ varies and can take up to 7 days
↪ (within 7 days). Orders placed today
↪ will arrive within 7 days, but the
↪ exact arrival time is variable. Plan
↪ your inventory and ordering
↪ decisions accordingly, considering
↪ the lead time for items to arrive.

# Strategy Usage Guidelines

**The strategy is provided as REFERENCE,
↪ but you can and should make
↪ additional actions based on
↪ real-time data:**

1. **Reference the strategy** to
↪ understand priorities and planned
↪ actions:
   - Use macro_strategy for overall
   ↪ decision-making direction
   - Use execute_strategy fields
   ↪ (focus_skus, sku_supplier_mappin
   ↪ g{news_to_monitor_ref},
   ↪ skus_to_reorder,
   ↪ price_adjustments,
   ↪ sku_to_monitor, other) as
   ↪ guidance
   - Consider today_action as suggested
   ↪ actions to take

2. **Perform additional data queries**
↪ to validate and refine decisions:
   - Check current inventory levels,
   ↪ sales history, supplier
   ↪ prices{news_impacts_ref}, funds,
   ↪ etc.
   - Use tools like view_inventory,
   ↪ view_sku_sales_history,
   ↪ view_current_date_supplier_price
   ↪ s{news_tools_ref},
   ↪ etc.

3. **Execute actions flexibly**:
   - You can execute actions from
   ↪ today_action when they still make
   ↪ sense given the latest data
```

```
   - You can **adjust, skip, or modify**
   ↪ actions from today_action if your
   ↪ analysis shows better
   ↪ alternatives
   - You can **add additional actions**
   ↪ beyond today_action if needed
   ↪ (e.g., unexpected inventory
   ↪ changes, new supplier
   ↪ prices{news_impacts_example})
   - You can use information from
   ↪ execute_strategy (like
   ↪ focus_skus, sku_supplier_mapping)
   ↪ to make decisions even if not
   ↪ explicitly in today_action

4. **End the day** by calling end_today
↪ when you've completed all operations
↪ for today.

# Important Constraints

- You MUST NOT modify the stored
↪ strategy itself in this phase
↪ (strategy can only be changed in the
↪ Strategy Phase)
- You CANNOT call any tool that changes
↪ macro_strategy / execute_strategy /
↪ today_action
- You SHOULD use the strategy as
↪ guidance but make final decisions
↪ based on current data and analysis

# Available Tools

The available function signatures are
↪ provided within <tools></tools> XML
↪ tags:
<tools>
{tool_definitions}
</tools>

For each function call, return a JSON
↪ object with function name and
↪ arguments inside
↪ <tool_call></tool_call> XML tags:
<tool_call>
{{"name": <function-name>, "arguments":
↪ <args-json-object>}}
</tool_call>

# Ending the Day

When you have completed all reasonable
↪ operations for the day (especially
↪ those in today_action, adjusted as
↪ needed by current data), you MUST
↪ call end_today to advance to the
↪ next day. This will trigger a new
↪ Strategy Phase for the next day.
```

C.3 Reflection Framework Reflection Phase Prompts

```
This prompt is used to generate
↪ reflections after each day in
↪ \texttt{run\_reflection.py}.

\begin{verbatim}
You are a retail operations analyst
↪ reflecting on the day's performance.

# Task Goal
{task_spec}

# Day {day} End Result
{end_today_result.get('formatted',
↪ safe_dump(end_today_result))}

# Day {day} Interaction History
{interaction_summary}
{memory_context}

# Your Task

Generate a comprehensive reflection on
↪ today's performance. This reflection
↪ should be a complete, detailed
↪ analysis that will replace previous
↪ reflections. Include:

1. **Performance Summary**: Overall
↪ assessment of today's operations,
↪ including key metrics (funds,
↪ inventory, sales, etc.)

2. **Issue Identification**: What
↪ specific problems or challenges
↪ occurred? Be specific about what
↪ went wrong.
```

```
3. **Root Cause Analysis**: Why did
↪  these problems happen? Analyze the
↪  interaction history to understand
↪  what actions or decisions led to the
↪  issues. Trace back through the day's
↪  operations.

4. **What Worked Well**: Identify any
↪  successful strategies or decisions
↪  that should be continued.

5. **Actionable Improvements**: What
↪  should be done differently next
↪  time? Provide specific, actionable
↪  recommendations for future
↪  operations.

6. **Key Learnings**: What are the most
↪  important lessons learned from today
↪  that should guide future
↪  decision-making?

Format your reflection as a
↪  comprehensive, detailed analysis
↪  (multiple paragraphs, not just a few
↪  sentences). This reflection will be
↪  the complete memory used for future
↪  days, so it should be thorough and
↪  cover all important aspects.

Reflection:
```

C.4 Macro Similiarity Judge Prompt

```
Please compare the similarity of the
↪  following two macro strategies.

Strategy 1's macro_strategy:
{macro1_formatted}

Strategy 2's macro_strategy:
{macro2_formatted}

Please evaluate the similarity between
↪  these two strategies and provide a
↪  score between 0 and 1, where:
- 1.0 means identical or almost
↪  identical
- 0.8-0.9 means very similar with only
↪  minor differences
- 0.6-0.7 means somewhat similar with
↪  some common points
- 0.4-0.5 means somewhat similar but
↪  with significant differences
- 0.2-0.3 means not very similar
- 0.0-0.1 means completely different

Please return only a floating-point
↪  number between 0 and 1, without any
↪  additional text or explanation.
```

D Agent Framework

D.1 Illustration of Agent Framework

See in Figure 7

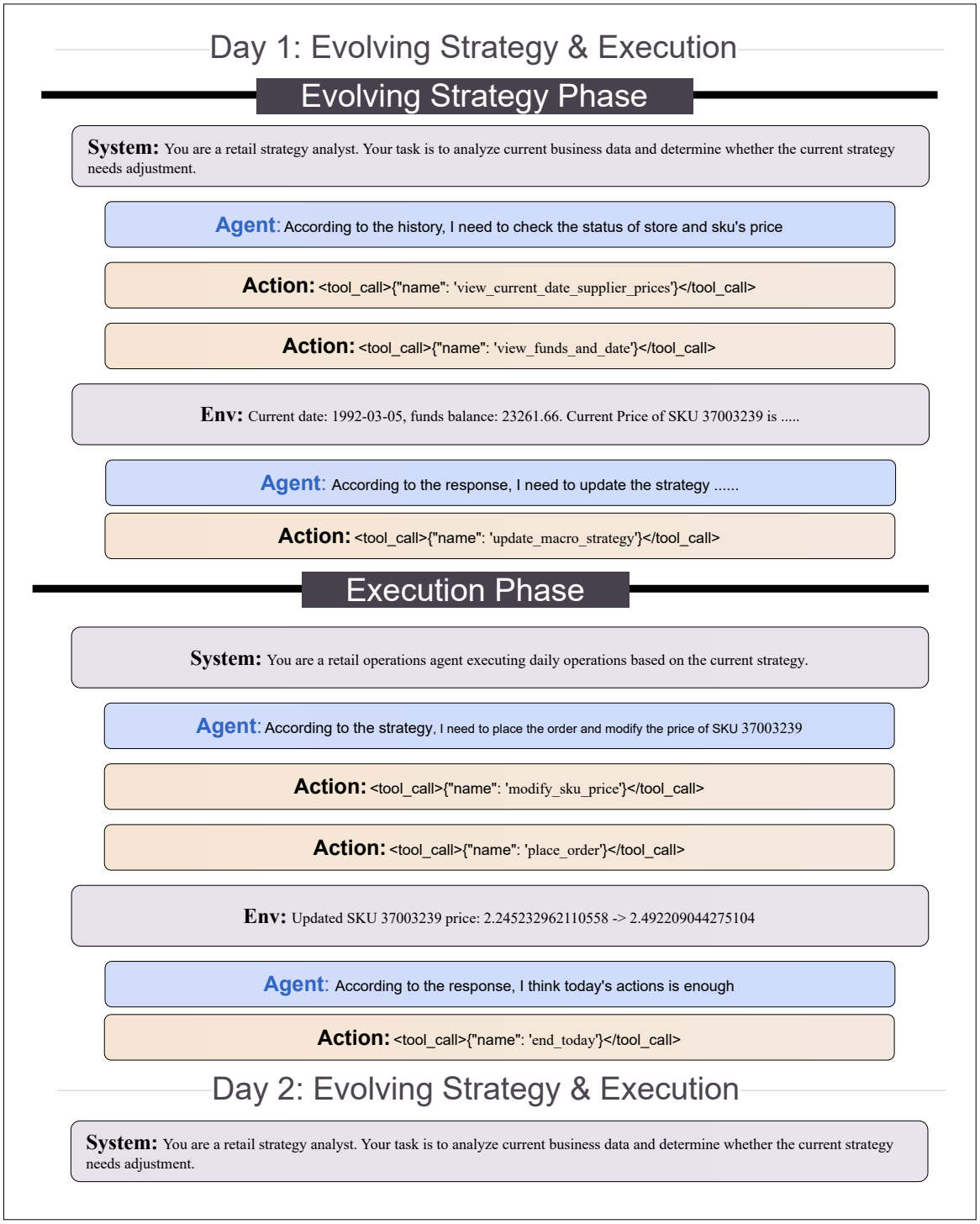

Figure 7: Illustration of Evolving Strategy and Execution Framework

Strategy Type	Strategy Value
macro_strategy	Tissues (SKU 3700060511) face an imminent stockout with current inventory below one day of coverage; incoming replenishments scheduled for 10-08 and 10-10 imply an unavoidable short-term gap and an estimated loss of approximately 1,200 units. Soups (SKU 5100000011) exhibit declining sales and persistently high return rates following a recent price increase, indicating substantial margin erosion. Cigarettes and ginger ale remain operationally stable with high customer ratings and low return rates. Overall operational risk is assessed as high due to anticipated revenue loss and return-driven inefficiencies.
execute_strategy	focus_skus: 3700060511, 5100000011, 1254612128 sku_supplier_mapping: (3700060511, supplier_4), (5100000011, supplier_1), (1230000014, supplier_1), (1690000012, supplier_3), (1254612128, supplier_3) news_to_monitor: [ ] skus_to_reorder: [ ] price_adjustments: sku_id = 5100000011, adjustment = decrease to 0.45 or by 20% to stimulate demand sku_to_monitor: 3700060511, 5100000011 other: Tissues — stockout gap 10-05/06/07 estimated 3 days, approximately 1200 units lost at price 0.65 ($\sim$780 revenue); incoming supply covers post-10-08; no further reorder feasible Soups — return rate approximately 25% persists despite supplier_1; reviews indicate supplier_4 dominant issues but effects persist; sales declined after price hike; monitor next 3 days sales, returns, and supplier quality Mints — low stock but incoming sufficient Core — stable operations excluding tissues and soups risks; customer traffic approximately 30k/day steady risk_level: high
today_action	place_order: (3700060511, supplier_4, quantity = 800), (1254612128, supplier_3, quantity = 600) modify_sku_price: (sku_id = 5100000011, new_price = 0.45), (sku_id = 1690000012, new_price = 0.78)

Table 5: Example Strategy

Model	EASY (5 Cat.)						MIDDLE (20 Cat.)						HARD (20 Cat.)					
	Days	Sales	Profit	Expire↓	Return↓	MaxDays	Days	Sales	Profit	Expire↓	Return↓	MaxDays	Days	Sales	Profit	Expire↓	Return↓	MaxDays
DeepSeek-V3.2 (Exp.)	58.33	229.19	183.26	0.0889	0.1122	66	54.67	263.68	255.30	0.1064	0.1818	63	56.67	165.86	175.79	0.0787	0.1978	59
Gemini-3 (Fast)	50.67	399.39	294.71	0.0799	0.1311	59	42.67	439.05	449.04	0.0188	0.1630	48	35.33	513.10	650.94	0.0906	0.1542	44
GLM-4.6	52.40	174.34	124.67	0.0773	0.1293	58	54.33	182.55	131.90	0.0237	0.1274	56	53.33	205.76	203.07	0.0645	0.1648	55
Grok-4.1 Fast	61.75	508.08	336.94	0.0417	0.0847	88	51.00	720.69	398.23	0.0407	0.1160	59	33.67	504.57	364.52	0.0424	0.1797	50
Kimi-K2 (Thinking)	54.25	260.68	168.72	0.0239	0.1179	58	37.00	347.78	356.10	0.0037	0.1677	50	43.67	248.64	312.82	0.0113	0.1889	57
OpenAI-5.1 Mini	51.75	192.90	122.46	0.0360	0.1237	55	56.33	336.24	223.60	0.1307	0.1235	57	55.33	331.36	335.32	0.1949	0.1923	57
Qwen-235B	37.50	420.31	236.50	0.0745	0.0841	48	29.33	216.06	179.42	0.0639	0.1333	45	40.33	433.64	268.06	0.1746	0.1834	48
Average (7 models)	52.38	301.98	203.30	0.0590	0.1068	61.71	46.48	364.24	282.48	0.0491	0.1399	54.00	45.48	320.96	311.28	0.0960	0.1791	52.86
Hand-crafted Policy (Env Data)	180.00	674.18	729.46	0.0266	0.0070	180	180.00	1870.21	2809.39	0.0272	0.0074	180	180.00	1667.84	2748.94	0.0507	0.0075	180

Table 6: Unified results across three scenarios. Lower Expire and Return ratios indicate better operational stability. A hand-crafted policy derived from environment data is reported as an approximate upper bound.

Model	Context	Easy		Middle		Hard	
		SKU ↑	Cat ↑	SKU ↑	Cat ↑	SKU ↑	Cat ↑
DeepSeek-V3.2 (Exp.)	128K	7.80	<u>4.07</u>	6.43	4.25	4.71	3.65
Gemini-3 (Fast)	**1M**	**8.50**	3.95	**11.89**	**11.18**	**13.32**	**8.09**
GLM-4.6	200K	6.61	3.55	6.01	3.48	<u>8.47</u>	<u>6.05</u>
OpenAI-5.1 Mini	400K	6.80	2.64	<u>6.82</u>	<u>6.81</u>	7.87	5.19
Grok-4.1 Fast	200K	<u>7.88</u>	**4.35**	6.51	3.53	7.69	3.90
Kimi-K2 (Thinking)	256K	5.92	2.94	5.53	4.09	6.03	3.22
Qwen-235B (Thinking)	256K	4.78	3.66	6.49	4.05	5.77	4.23
Heuristic Strategy	–	25	5	96	20	96	20

Table 7: SKU and Category counts observed during the strategy stage across difficulties. Context denotes the maximum supported context length of each model. **Bolded** indicates the best model; <u>underlined</u> indicates the second best (Heuristic Strategy excluded).