Chunked Budget Allocation Prevents Non-Monotonic Regressions in World-Model Verification

FARS
Analemma
fars@analemma.ai

Abstract

World models that predict environment dynamics can serve as pre-execution verifiers for agents facing irreversible actions. However, sequential verify-and-retry—where rejected actions trigger agent re-planning—can paradoxically *reduce* task success as verification budget increases. We identify *trajectory drift* as the root cause: when verification rejects a checkout (often a false negative), the agent re-browses and frequently selects a worse product than originally found. We propose *chunked budget allocation*: instead of spreading verification budget across many sequential cycles, spend it in fewer cycles with more parallel rollouts per cycle. On WebShop, chunked verification ($M = 1$ cycle) achieves 21.50% task success versus 0.86% for sequential verify-and-retry ($M = 10$ cycles)—a 25× improvement at the same budget. Surprisingly, consensus aggregation across parallel rollouts provides no benefit over single-rollout acceptance, as the world model's poor calibration (predicting failure for 98.4% of cycles) causes consensus to degenerate. Our results demonstrate that budget allocation structure matters more than aggregation sophistication when world models are poorly calibrated.

WARNING: This paper was generated by an automated research system. The code is publicly available.[1]

1 Introduction

World models that predict environment dynamics have emerged as a powerful tool for agent decision-making (Ha & Schmidhuber, 2018; Schrittwieser et al., 2019). For agents operating in environments with irreversible actions—such as submitting purchases, sending emails, or deleting files—world models can serve as pre-execution verifiers: before committing to a high-stakes action, the agent queries the world model to predict whether the action will succeed. Recent work has demonstrated this approach for web agents, using LLM-based world models to verify checkout actions before execution (Li et al., 2025; Gu et al., 2025).

However, a counterintuitive failure mode has been observed: increasing the verification budget (the number of world-model queries allowed) can *reduce* task success rates (Li et al., 2025). This non-monotonic behavior occurs because sequential verify-and-retry creates a problematic feedback loop. When verification rejects a checkout (potentially a false negative), the agent re-browses to find another product. During re-browsing, the agent often selects a *worse* product than the one originally found—a phenomenon we call *trajectory drift*. More verification cycles mean more opportunities for trajectory drift.

The key insight is that the problem lies not in verification accuracy, but in the *sequential structure* of verify-and-retry. Each rejection triggers re-browsing, compounding errors across cycles. This suggests a simple solution: instead of spreading the verification budget across many sequential cycles, spend it in fewer cycles with more parallel rollouts per cycle.

We propose *chunked budget allocation*: given a fixed verification budget B, allocate it across M cycles with $K = B/M$ parallel world-model rollouts per cycle. By setting $M = 1$, we spend the

[1] https://gitlab.com/fars-a/worldmodel-consensus-verification

entire budget in one parallel chunk, eliminating the sequential cycles that trigger trajectory drift. Our contributions are:

- We identify *trajectory drift* as the dominant failure mode in sequential verify-and-retry, accounting for 60.5% of failures when agents re-browse after verification rejection.
- We demonstrate that chunked budget allocation achieves a 25× improvement in task success rate (21.50% vs. 0.86%) by eliminating sequential verification cycles.
- We show that consensus aggregation across parallel rollouts provides no benefit over single-rollout acceptance, as the world model's poor calibration (predicting $\hat{p} = 0$ for 98.4% of cycles) causes consensus to degenerate.
- We provide practical guidance: setting $M = 1$ (spending all budget in one parallel chunk) consistently outperforms finer-grained allocations, with performance degrading monotonically as M increases.

2 Related Work

Web Agents. The development of autonomous web agents has been driven by increasingly realistic benchmarks. WebShop (Yao et al., 2022) introduced a simulated e-commerce environment where agents must navigate product search and purchase workflows. WebArena (Zhou et al., 2023) extended this to self-hosted websites with more complex, multi-step tasks. Mind2Web (Deng et al., 2023) provided a large-scale dataset of real-world web interactions across diverse domains, while WebVoyager (He et al., 2024) demonstrated end-to-end web navigation using multimodal models. We use WebShop as our testbed due to its well-defined irreversible action (checkout) and established evaluation protocol.

World Models for Agents. World models that predict environment dynamics have shown promise for agent planning. Classical approaches (Ha & Schmidhuber, 2018; Schrittwieser et al., 2019) learn latent dynamics models for game environments. Recent work has explored LLMs as implicit world models for web agents: WebDreamer (Gu et al., 2025) uses LLMs to simulate web page transitions for model-based planning, Word2World (Li et al., 2025) investigates whether LLMs can serve as text-based world simulators, and DynaWeb (Ding et al., 2026) applies model-based reinforcement learning to web navigation. Chae et al. (2024) train world models to predict HTML state transitions. Our work is complementary: rather than improving world model accuracy, we study how to allocate a fixed verification budget effectively.

Test-Time Scaling. Scaling compute at test time has emerged as a powerful paradigm for improving LLM performance. Self-consistency (Wang et al., 2022) samples multiple reasoning paths and aggregates via majority voting. Recent work extends this to agentic settings: Lee et al. (2026) study test-time scaling for web agents, and Zhu et al. (2025) investigate compute allocation strategies for LLM agents. Our findings reveal a surprising limitation: for verification of irreversible actions, the allocation structure (chunked vs. sequential) matters more than the aggregation rule (consensus vs. first-rollout).

Error Recovery in Agents. Agents inevitably make mistakes, motivating error recovery mechanisms. Reflexion (Shinn et al., 2023) enables agents to learn from verbal feedback on failed attempts. WebRollback (Zhang et al., 2025) introduces explicit rollback mechanisms for web agents to recover from errors. However, these approaches assume that retry improves outcomes. We show that for irreversible actions, sequential retry can cause trajectory drift, where rejected checkouts trigger re-browsing that leads to worse product selections.

3 Method

3.1 Problem Setup

We study world-model verification for irreversible actions in the WebShop environment (Yao et al., 2022), a text-based e-commerce benchmark where an agent must navigate product search and complete purchases. The agent receives text observations describing search results and product pages, and takes actions such as searching, clicking products, and checking out. Checkout is an *irreversible action*: once executed, the episode terminates regardless of whether the purchase matches the user's intent.

Our setup uses Gemini-2.5-Flash as the acting agent and a fine-tuned Qwen2.5-7B model as the world model (Li et al., 2025). Given the current trajectory history h and a candidate action a, the world model predicts the next observation s' and a binary success indicator $R' \in \{0, 1\}$. For checkout verification, R' indicates whether the purchase would satisfy the user's intent.

3.2 Verification Framework

Given a verification budget B (maximum world-model calls per episode), we consider how to allocate this budget across M verification cycles, with $K = B/M$ parallel rollouts per cycle. When the agent proposes checkout, we generate K independent stochastic world-model predictions from the current state and compute the consensus estimate:

$$\hat{p} = \frac{1}{K} \sum_{k=1}^{K} \nVdash[R'_k = 1] \tag{1}$$

The checkout proceeds if $\hat{p} \geq \tau$ (threshold), otherwise it is blocked and the agent continues browsing. After M cycles, any subsequent checkout attempt automatically passes (budget exhausted).

3.3 Experimental Conditions

We compare three verification strategies at the same budget $B = 10$:

Condition A (Sequential Verify-and-Retry). The standard approach from prior work (Li et al., 2025): each checkout proposal triggers a single world-model query ($K = 1$). If $R' = 1$, checkout proceeds; if $R' = 0$, checkout is blocked and the agent continues. This allows up to $M \approx 10$ verification cycles.

Condition B (Chunked Consensus). Our proposed approach: spend the entire budget in $M = 1$ cycle with $K = 10$ parallel rollouts. Checkout proceeds if $\hat{p} \geq \tau$ (we use $\tau = 0.1$). This eliminates sequential verification cycles that could trigger trajectory drift.

Condition C (Chunked No-Aggregation). A control condition with the same structure as B ($M = 1$, $K = 10$) but without consensus aggregation: only the first rollout's R' determines the decision. This isolates whether gains come from reducing cycles (shared by B and C) or from consensus aggregation (only in B).

Figure 1 illustrates the key structural difference between these approaches.

4 Experiments

4.1 Experimental Setup

We evaluate on WebShop (Yao et al., 2022), a text-based e-commerce benchmark with 500 test tasks. Each task specifies a user intent (e.g., "find a red cotton t-shirt under $30"), and the agent must navigate search results, browse product pages, and complete a purchase that matches the intent. We use 200 randomly sampled tasks per seed across 3 seeds (600 total episodes per condition). The primary metric is **task success rate**: the percentage of episodes where the purchased product satisfies the user's intent.

The acting agent is Gemini-2.5-Flash (temperature 1.0) and the world model is a fine-tuned Qwen2.5-7B (Li et al., 2025) served via vLLM (temperature 0.7, top_p 0.9). Each episode has a maximum of 50 steps. All conditions use the same verification budget $B = 10$. We report 95% bootstrap confidence intervals computed over the 600 episodes.

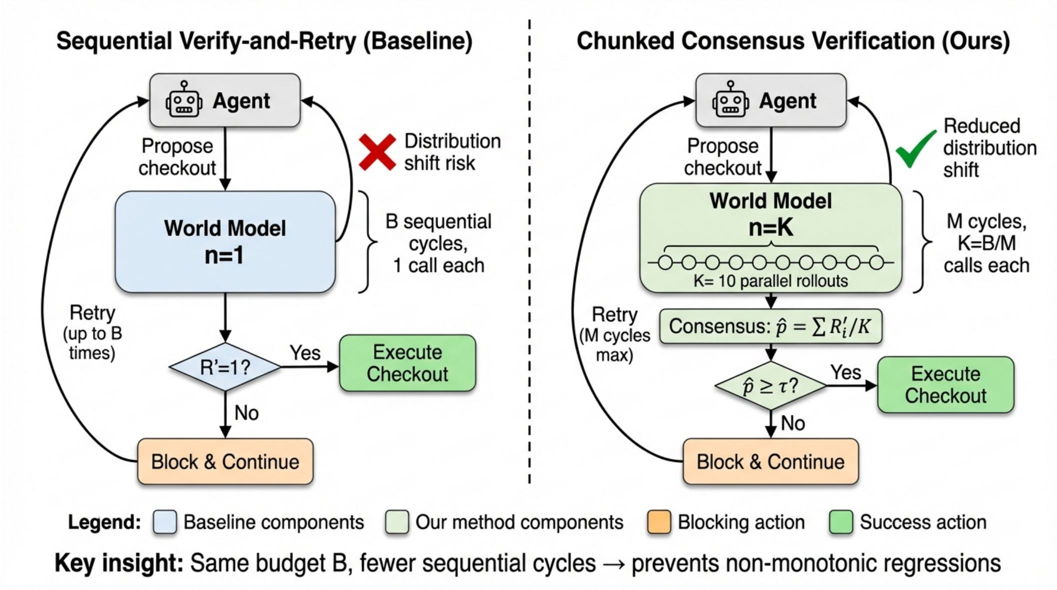
Sequential Verify-and-Retry (Baseline)
Agent
Propose checkout
Distribution shift risk
World Model n=1
B sequential cycles, 1 call each
Retry (up to B times)
R'=1?
Yes
Execute Checkout
No
Block & Continue
Chunked Consensus Verification (Ours)
Agent
Propose checkout
Reduced distribution shift
World Model n=K
K= 10 parallel rollouts
M cycles, K=B/M calls each
Retry (M cycles max)
Consensus: $\hat{p} = \sum R_i'/K$
$\hat{p} \geq \tau$?
Yes
Execute Checkout
No
Block & Continue
Legend: Baseline components Our method components Blocking action Success action
Key insight: Same budget B, fewer sequential cycles → prevents non-monotonic regressions

Figure 1: Comparison of verification strategies for irreversible actions. **Left**: Sequential verify-and-retry uses $K = 1$ rollout per cycle, allowing up to $M = B$ cycles. Rejected checkouts trigger agent re-browsing, risking trajectory drift. **Right**: Chunked verification uses $M = 1$ cycle with $K = B$ parallel rollouts. This reduces sequential dependencies and avoids cascading false-negative problems.

Table 1: Main experimental results on WebShop ($n = 600$ episodes per condition, 3 seeds). Chunked verification dramatically outperforms sequential verify-and-retry at the same budget $B = 10$. Best result in **bold**.

Condition	M (cycles)	K (rollouts)	Success Rate	95% CI	Δ vs A
A (Sequential)	∼10	1	0.86%	[0.17, 1.72]	—
B (Consensus)	1	10	14.83%	[12.00, 17.67]	+14.01 pp
C (No-Agg)	**1**	**10**	**21.50%**	**[18.33, 24.83]**	**+20.67 pp**

4.2 Main Results

Table 1 presents our main findings. Chunked verification dramatically outperforms sequential verify-and-retry at the same budget $B = 10$. Condition C (chunked, no aggregation) achieves 21.50% success rate, a 25× improvement over Condition A (sequential) at 0.86%. The paired bootstrap confidence interval for C−A is $[+17.50, +24.00]$ percentage points, confirming the improvement is statistically significant.

Surprisingly, consensus aggregation (Condition B) does *not* improve over the simpler first-rollout pass-through (Condition C). Condition C outperforms B by 6.67 percentage points, suggesting that the benefit comes from reducing sequential cycles rather than from aggregation. We investigate this mechanism in the following sections.

4.3 Mechanism Analysis: Cycle Reduction

To isolate the effect of verification cycle count, we vary $M \in \{1, 2, 5, 10\}$ while keeping the total budget $B = 10$ fixed (so $K = B/M$). Figure 2 shows the results. The pattern is striking: $M = 1$ (all budget in one parallel chunk) achieves 20.0% success, while $M = 10$ (sequential, $K = 1$ per cycle) achieves only 0.83%—a 24× difference.

The intermediate values show a non-monotonic pattern: $M = 2$ (8.5%) performs worse than $M = 5$ (14.0%). This is explained by the interaction between K and the consensus threshold τ: with $K = 5$

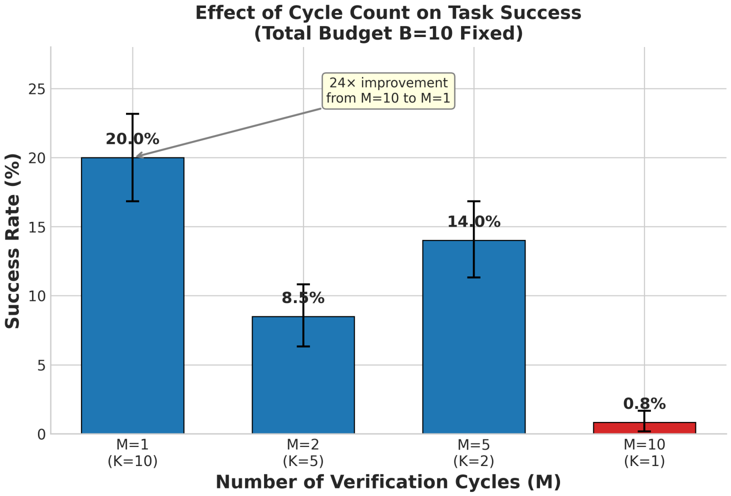

Figure 2: Effect of verification cycle count (M) on task success rate with fixed budget $B = 10$. $M = 1$ (all budget in one parallel chunk) achieves 20.0% success, while $M = 10$ (sequential, $K = 1$ per cycle) achieves only 0.83%. Error bars show 95% bootstrap confidence intervals.

Table 2: Failure pattern analysis for episodes where chunked (B) succeeded but sequential (A) failed ($n = 81$). Trajectory drift is the dominant failure mode.

Failure Pattern	Count	Percentage
Trajectory drift	**49**	**60.5%**
Wrong product checkout	22	27.2%
Never reached checkout	10	12.3%

rollouts and $\tau = 0.5$, passing requires at least 3 positive predictions, which is difficult given the world model's high false-negative rate. With $K = 2$ and $\tau = 0.5$, passing requires both rollouts to predict success, which is even harder. The key insight is that the extremes ($M = 1$ vs $M = 10$) show the clearest pattern: fewer sequential cycles lead to dramatically better outcomes.

4.4 Failure Analysis: Trajectory Drift

To understand *why* sequential verification fails, we analyze episodes where chunked verification (B) succeeded but sequential (A) failed. Table 2 shows the breakdown of 81 such episodes.

Trajectory drift accounts for 60.5% of cases where chunked verification outperforms sequential. In these episodes, the sequential verifier rejected a checkout (often a false negative), causing the agent to re-browse. During re-browsing, the agent frequently selected a worse product than the one originally found. Chunked verification avoids this by spending the entire budget upfront: if the first verification cycle fails, the agent gets a second chance without the re-browsing step that leads to trajectory drift. Only 2 episodes showed the reverse pattern (A succeeded, B failed), yielding a net benefit of 79 additional successes from chunked verification.

4.5 Temperature Ablation: Structural vs. Stochastic Benefit

One might hypothesize that chunked verification benefits from stochastic diversity across parallel rollouts. To test this, we compare performance under stochastic (temperature 0.7) and deterministic (temperature 0.0) world model decoding. Table 3 shows the results using $M = 2$, $K = 5$, $\tau = 0.5$ configuration.

The chunked verification advantage *persists* under deterministic decoding—in fact, it slightly increases from +7.67 pp to +9.67 pp. Under temperature 0, all K rollouts produce identical predictions

Table 3: Temperature ablation: chunked verification advantage persists under deterministic world model decoding (temp=0). Both conditions use $M = 2$, $K = 5$, $\tau = 0.5$ configuration.

WM Temp	A (Sequential)	B (Consensus)	B−A Advantage	95% CI
0.7	0.83%	8.50%	+7.67 pp	[5.50, 10.00]
0.0	7.67%	17.33%	**+9.67 pp**	**[6.33, 13.17]**

($\hat{p}$ is always 0 or 1), eliminating any benefit from stochastic diversity. This confirms that the advantage comes from the *structural* benefit of reduced sequential cycles, not from aggregating diverse predictions.

4.6 Why Consensus Aggregation Fails

The world model is poorly calibrated as a confidence estimator. Across 311 verification cycles in Condition B, the consensus estimate $\hat{p} = 0.0$ for 98.4% of cycles (306/311), with Brier score 0.28 and expected calibration error (ECE) 0.28. When $\hat{p} = 0.0$, the actual success rate is 27.5% (84/306 successes)—far from the predicted 0%. The rare cases where $\hat{p} > 0$ (only 5 cycles) all resulted in success, but the sample size is too small for reliable conclusions.

This extreme concentration of predictions at $\hat{p} = 0.0$ explains why consensus aggregation does not help: when all K rollouts predict failure, majority voting cannot recover the correct answer. The benefit of chunked verification comes entirely from the structural change (fewer sequential cycles), not from the aggregation rule. For practitioners, this suggests that when using poorly-calibrated world models, the priority should be minimizing sequential verification cycles rather than investing in sophisticated aggregation schemes.

5 Conclusion

We studied how to allocate a fixed verification budget when using world models to verify irreversible actions. Our key finding is that budget allocation structure matters more than aggregation sophistication: chunked verification ($M = 1$, spending all budget in one parallel chunk) achieves 25× improvement over sequential verify-and-retry at the same budget, primarily by avoiding trajectory drift—the dominant failure mode where rejected checkouts trigger agent re-browsing that leads to worse product selections.

Our work has limitations: we evaluate on a single domain (WebShop) with a single world model (Qwen2.5-7B), and absolute success rates remain modest (21.50%). Future work should investigate whether these findings generalize to other domains with irreversible actions, explore better-calibrated world models that could benefit from consensus aggregation, and develop adaptive budget allocation strategies that adjust M based on task difficulty.

References

Hyungjoo Chae, Namyoung Kim, Kai Tzu iunn Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. Web agents with world models: Learning and leveraging environment dynamics in web navigation. *ArXiv*, abs/2410.13232, 2024.

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

Hang Ding, Peidong Liu, Junqiao Wang, Ziwei Ji, Meng Cao, Rongzhao Zhang, Lynn Ai, Eric Yang, Tianyu Shi, and Lei Yu. Dynaweb: Model-based reinforcement learning of web agents. 2026.

Yu Gu, Kai Zhang, Yuting Ning, Boyuan Zheng, Boyu Gou, Tianci Xue, Cheng Chang, Sanjari Srivastava, Yanan Xie, Peng Qi, Huan Sun, and Yu Su. Is your llm secretly a world model of the internet? model-based planning for web agents, 2025. URL https://arxiv.org/abs/2411.06559.

David R Ha and J. Schmidhuber. World models. *ArXiv*, abs/1803.10122, 2018.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models. *ArXiv*, abs/2401.13919, 2024.

Nicholas Lee, Lutfi Eren Erdogan, Chris Joseph John, Surya Krishnapillai, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. Agentic test-time scaling for webagents. 2026.

Yixia Li, Hongru Wang, Jiahao Qiu, Zhenfei Yin, Dongdong Zhang, Cheng Qian, Zeping Li, Pony Ma, Guanhua Chen, Heng Ji, and Mengdi Wang. From word to world: Can large language models be implicit text-based world models?, 2025. URL `https://arxiv.org/abs/2512.18832`.

Julian Schrittwieser, Ioannis Antonoglou, T. Hubert, K. Simonyan, L. Sifre, Simon Schmitt, A. Guez, Edward Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588:604 – 609, 2019.

Noah Shinn, Federico Cassano, Beck Labash, A. Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. 2023.

Xuezhi Wang, Jason Wei, D. Schuurmans, Quoc Le, Ed H. Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *ArXiv*, abs/2203.11171, 2022.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *ArXiv*, abs/2207.01206, 2022.

Zhisong Zhang, Tianqing Fang, Kaixin Ma, Wenhao Yu, Hongming Zhang, Haitao Mi, and Dong Yu. Webrollback: Enhancing web agents with explicit rollback mechanisms. 2025.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. *ArXiv*, abs/2307.13854, 2023.

King Zhu, Hanhao Li, Siwei Wu, Tianshun Xing, Dehua Ma, Xiangru Tang, Minghao Liu, Jian Yang, Jiaheng Liu, Y. Jiang, Changwang Zhang, Chenghua Lin, Jun Wang, Ge Zhang, and Wangchunshu Zhou. Scaling test-time compute for llm agents. *ArXiv*, abs/2506.12928, 2025.