AgenticSTS: A Bounded-Memory Testbed for Long-Horizon LLM Agents

Xiangchen Cheng[1,2], **Yunwei Jiang**[2,3], **Jianwen Sun**[1,3,4], **Zizhen Li**[1,3,4], **Chuanhao Li**[1], **Xiangcheng Cao**[1], **Yihao Liu**[1], **Fanrui Zhang**[3,5], **Li Jin**[2], **Kaipeng Zhang**[1,†]

[1]Alaya Lab [2]Shanghai Jiao Tong University [3]Shanghai Innovation Institute [4]Nankai University [5]University of Science and Technology of China
[†]Corresponding author

Memory for a long-horizon LLM agent is a contract about what each future decision is allowed to see. The simplest contract appends past observations, tool calls, and reflections to every prompt, which makes prior context easy to access but also turns it into a jumbled mixture in which the effect of any single memory component is hard to isolate. We introduce and instrument an alternative bounded contract: every decision is made from a fresh user message assembled by typed retrieval, with no raw cross-decision transcript appended. The prompt thus stays bounded across runs of any length, and any single layer can be ablated in isolation. We instantiate the contract in Slay the Spire 2, a closed-rule stochastic deck-building game whose runs require hundreds of tactical and strategic decisions. A public online benchmark of frontier LLMs on the same game reports zero wins at the lowest difficulty across five configurations, and the developer-reported human win rate at the same difficulty is 16%; the task is hard but not saturated. Within our harness, a fixed-A_0 ablation shows the largest observed difference when triggered strategic skills are enabled: the no-store baseline wins 3/10 games and adding the skill layer 6/10. At this sample size the comparison is directional rather than statistically decisive (Fisher exact $p \approx 0.37$); a cross-backbone probe and public accumulating-context baselines are reported as operational comparisons rather than controlled tests of the contract variable itself. We release a reproducible testbed: 298 completed trajectories with condition tags, frozen memory/skill snapshots, prompt records, and analysis scripts—an agent design and a validated, reusable methodology for studying how explicit memory layers shape long-horizon LLM-agent decisions.

Project page: https://github.com/AlayaLab/AgenticSTS
Code: https://github.com/AlayaLab/AgenticSTS
Data: https://huggingface.co/datasets/ShandaAI/AgenticSTS-trajectories
Correspondence: kaipeng.zhang@shanda.com
Date: July 3, 2026

ALAYA Lab

1 Introduction

For a long-horizon LLM agent, memory is not a place to store text; it is a contract about what each future decision is allowed to see. One common contract appends past observations, tool calls, and reflections to the next prompt [47, 28, 36]. Another distills prior experience into typed records and retrieves only the pieces selected for the current decision [32, 25, 46, 17, 27]. This choice is not just an engineering detail: it determines what evidence the model sees, what stale information can re-enter a decision, and which component can be ablated when the agent succeeds or fails. We therefore ask whether a long-horizon benchmark [29, 18] can make the memory interface bounded, inspectable, and reusable rather than treating context growth as an implicit default. Practitioners increasingly frame this through the *agent loop* and its context or "loop" engineering [2, 15]—the per-iteration choice of which prior experience to place in a finite window; our bounded contract is a formal, ablatable answer to the memory stage of that loop.

We instantiate this question in Slay the Spire 2, a roguelike deck-building game. A run is a stochastic strategic campaign: the agent chooses map routes, fights turn-based battles, drafts cards, buys or skips

Figure 1 Overview of our paper: a bounded, typed memory contract turns long-horizon LLM-agent memory into an ablatable evaluation surface. Summary performance labels in this schematic (e.g. relative scores and ladder reach) are illustrative; the exact numbers, denominators, and caveats—including that cross-agent comparisons are operational rather than matched ablations and that win-rate differences are directional at our sample size—are given in §6–7.

items, and preserves scarce health over many delayed consequences. The rule space is closed, symbolic, and text-readable, so game facts and legal actions can be supplied as structured records rather than pixels. At the same time, success requires hundreds of local and long-range decisions under random card draws, rewards, enemies, events, and difficulty modifiers. Public benchmarks indicate that the task is not already saturated. The game ships an ascension difficulty system that runs from A_0 (the easiest tier) up to A_{10} (the hardest). At A_0, a publicly available benchmark of frontier LLMs on this game reports zero wins across five model configurations, and the developer-reported human win rate is 16% [1, 23]. Together these numbers make Slay the Spire 2 a hard but unsaturated testbed for studying long-horizon LLM-agent memory.

Our agent, AgenticSTS, implements typed retrieval as a bounded memory contract. For every decision, the user message is freshly composed from five slots: fixed protocol instructions (L_1), state-specific schemas and legal action formats (L_2), retrieved game rules (L_3), episodic summaries (L_4), and triggered strategic skills (L_5). The slots differ in mutability and experimental role: L_1 and L_2 are fixed; L_3 can be filtered; L_4 and L_5 can be disabled, frozen, or made writable between runs through postrun analysis; raw cross-decision transcripts are not appended. The resulting prompt interface turns memory from "how much history fits" into "which typed evidence is selected," a form that can be inspected and re-aggregated across conditions.

Across 298 completed trajectories, we run two main evidence streams and a cross-backbone probe. The first holds difficulty fixed at A_0 and varies the five memory slots: a no-scaffold baseline wins 3/10 games (Wilson 95% confidence interval $[10.8\%, 60.3\%]$, i.e., the true win-rate is likely between 10.8% and 60.3%), and the largest observed difference coincides with enabling L_5 skills, which reach 6/10 in each scaffolded cell ($[31.3\%, 83.2\%]$). At this sample size the difference is directional rather than statistically decisive

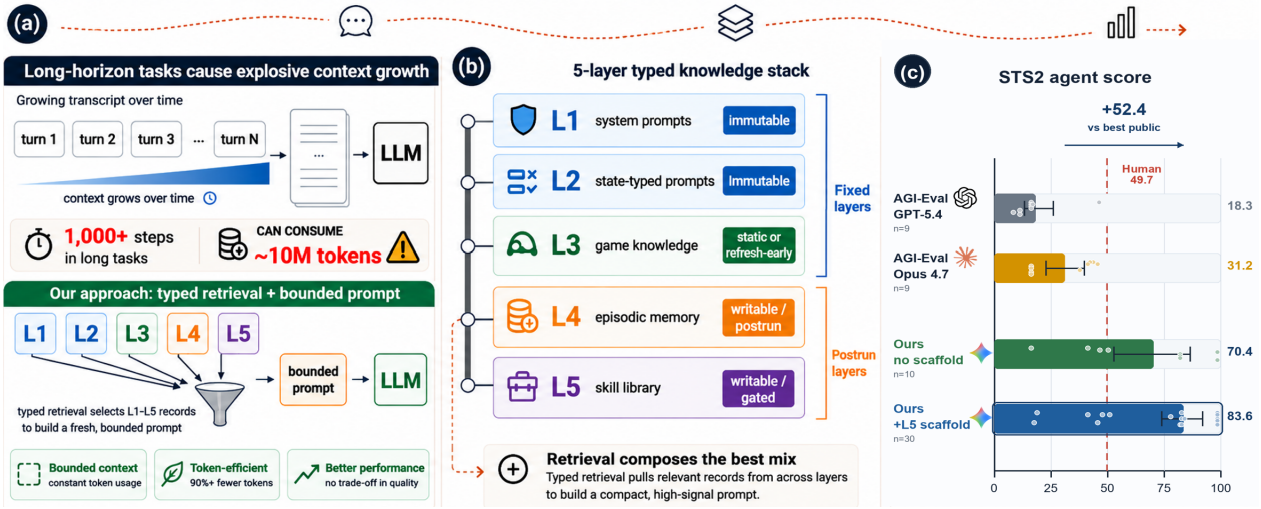

Figure 2 Typed retrieval as a bounded-memory contract: **(a)** per-decision composition, **(b)** the five typed layers, and **(c)** AgenticSTS scores vs AGI-Eval rows and the community human reference [31]. In-panel summary labels (e.g. "no trade-off", token-efficiency and score-gap figures) are illustrative; cross-backbone and external comparisons are operational context, not matched ablations of the memory contract, and within-harness win-rate differences are directional at our sample size—see §6–7 for exact numbers and caveats.

(Fisher exact $p \approx 0.37$), and the five typed slots remain individually ablatable. The second stream climbs the difficulty ladder with L_4 and L_5 stores that update between runs, reaching high-difficulty probes at A_6–A_8. A cross-backbone probe (the underlying LLM is swapped among Gemini, Qwen, and DeepSeek) exercises the same ablation surface on other model families; the frozen stack is backbone-sensitive, with model-specific caveats detailed in §6.4.

The released archive supports re-aggregation and cross-condition re-analysis: a reader can recompute the headline fixed-A_0 cells or slice trajectories by condition tag using the included scripts. We ship the condition-tagged trajectories, the SHA-anchored L_4+L_5 snapshots used in the fixed-A_0 matrix, decision-time prompt records, and Wilson/bootstrap analysis scripts.

Contributions. We contribute (see Figure 1):

(i) a per-decision composition interface assembling prompts from typed L_1–L_5 slices rather than a raw transcript;
(ii) evidence that, under the bounded contract, the largest observed A_0 difference coincides with enabling triggered L_5 skills ($3/10 \rightarrow 6/10$, directional at this sample size rather than statistically significant), that the typed slots remain individually ablatable across three model backbones, and that the surface admits ladder probes at high-difficulty A_6–A_8;
(iii) a reusable archive of 298 trajectories with condition tags, SHA-anchored L_4/L_5 snapshots, prompts, and Wilson/bootstrap scripts for community study of context use in long-horizon agents.

Claims concern layer separability inside the contract; a matched accumulating-context comparison is future work (Limitations). Prior prompt-history [47, 28, 36], structured-memory [25, 7, 17, 44], and skill-library agents [20, 24, 51] motivate the design space. §3 presents the testbed; §4 the contract; §5 the protocol; §6 the results.

2 Related Work

We build on four recent research threads—prompt-history agents, externalized memory, skill libraries, and long-horizon game testbeds—and target their joint problem: which slice of prior experience enters each decision, and whether that route can itself be ablated.

Loop and context engineering. In practice, agent design is increasingly framed as *loop engineering*: specifying

Table 1 Positioning map for prior-experience interfaces. Green checkmarks mark a central axis; orange triangles mark a partial or optional axis. Citations appear in the surrounding paragraphs.

Family	Transcript	Typed memory	Skills	Layer ablation	Game policy	Gap
Prompt-history / replay	✓	–	▲	–	▲	attribution
Structured memory	–	✓	–	▲	–	policy
Skill-library agents	▲	▲	✓	▲	▲	contract
Ours	–	✓	✓	✓	✓	**joint test**

a goal, tools, termination, and—centrally—the memory and context policy of a control loop that runs over hundreds of steps and multiple sessions [2]. The recurring failure mode is context growth: appending full transcripts and tool logs to every call overflows the window and dilutes attention, whereas "token-poor" loops keep only the last few messages or short summaries and therefore depend on an explicit memory store [15]. This is exactly the axis our contract isolates. That literature is largely qualitative and centered on coding agents; the academic threads below supply the mechanisms, and we make the memory stage of the loop a typed, bounded, ablatable contract on a hard long-horizon game.

Long-horizon LLM agents and prompt-visible history. ReAct and Reflexion made it natural for observations and self-critiques to reappear in later LLM calls [47, 28]. Recent horizon analyses [29, 41, 39, 18] examine how small errors compound over many steps.

Typed and structured memory. MemGPT, Mem0, MemoryOS, GAM, hierarchical procedural memory, and Agent Workflow Memory [25, 7, 17, 44, 10, 42] move information out of raw message history into external stores. Adjacent typed-memory frameworks [32, 27, 46] group memory by capacity; we instead role-type slots by mutability and retrieval source. Most evaluation is in dialogue or QA; in our setting retrieval feeds an action policy in a stochastic environment.

Self-evolving skill libraries. Voyager pioneered an external library of agent-written skills [36]; SkillsBench, SkillOS, Memento-Skills, SAGE/SkillRL, SkillWeaver, ExpEL, and DyStIL extend the design [20, 24, 51, 37, 50, 49, 35]. In the SoK notation $S = (C, \pi, T, R)$ [16], our L_5 guides correspond to (C, π): a trigger selects a prose policy for the next decision.

LLM agents on games. Games such as Crafter, NetHack, BALROG, LMGame-Bench, DSGBench, Gameverse, and RAGEN [12, 19, 26, 14, 33, 48, 40] provide stochastic testbeds for agents. Card-game work includes end-to-end policy networks [45], LoRA-tuned draft models [4], cross-card-game LLM evaluation [38], and LLM play on the original Slay the Spire with simplified rule sets [3, 8].

Public Slay the Spire 2 LLM agents. The public Slay the Spire 2 ecosystem—STS2MCP, HermesBridge, AI-Spire, CharTyr [11, 13, 5, 6]—does not report a matched ablation over prompt strictness, episodic memory, and triggered skills. AGI-Eval [1] lists zero A_0 victories across five frontier-model rows with mixed denominators.

We combine these threads in one bounded-memory contract, so future work can compare alternative contracts under the same harness.

3 The Slay the Spire 2 Testbed

Slay the Spire 2 is a turn-based deck-building roguelike: an agent builds a deck during a run, fights stochastic battles, chooses routes and rewards, and climbs an ordinal difficulty ladder. The game is useful for evaluating LLM-agent memory because it is long-horizon but not visually opaque. Rules, cards, relics, enemies, events, legal actions, and state transitions can be represented as text records, while success still requires sustained planning across many contingent decisions. We release the resulting runs as a reusable evaluation resource (§5.4).

3.1 Four properties of Slay the Spire 2 as an LLM testbed

(*P1*) Closed, enumerable, LLM-readable rule space. Public database snapshots index hundreds of typed records (576 cards, 293 relics, 115 monsters, 87 encounters, and 66 events in Spire Codex's May 2026 API; these are database counts, not unique experimental-patch counts) [30]. Unlike pixel-rendered Crafter [12] or the inherited code complexity of NetHack [19], Slay the Spire 2 has a compact rule space that can be loaded into a typed knowledge layer (§4). This makes L_3 part of the evaluation substrate rather than a hidden source of game knowledge.

(*P2*) Empirically long horizon. A typical run lasts a median ∼80 min wall-clock (IQR 37–109 min) and contains 67 LLM strategic calls (IQR 27–105). Roughly 500 additional per-run decisions, such as combat targets, treasure choices, map nodes, and hand selection, are mechanically resolved or routed to a fast tier with bounded combat context (§4). This is precisely the regime where message-history accumulation becomes costly relative to typed recomposition [29, 39, 18].

(*P3*) Multi-axis stochasticity. Random card draw, shuffle order, reward offerings, map paths, relic effects, elite and event placements, and Ascension modifiers prevent simple trajectory replay. A strong policy must generalize over states, not memorize a fixed route.

(*P4*) State-conditioned combat math. Damage and status pipelines combine hand contents, enemy intent, block timing, and effects such as *vulnerable*, *weak*, *strength*, and *dexterity*. The agent must compute from the current state; web-like recall is much less useful than state-conditioned calculation.

3.2 Ascension ladder and scoring

The 11-level Ascension ladder (A_0–A_{10}, with A_{10} the maximum) gives the testbed an ordinal difficulty scale. Higher Ascensions stack modifiers that change strategic priorities, so climbing the ladder is not just repeating A_0 with larger numbers. We therefore use two complementary evidence streams: a fixed-A_0 matrix that isolates components at one difficulty (§5) and an auto-mode ladder in which the agent advances after victories and retries after defeats (§5).

Runs are scored with a derived analysis score:

$$s = \begin{cases} 100 & \text{if victory,} \\ \text{floor} + \frac{52}{3} \cdot \text{bosses} & \text{otherwise,} \end{cases} \tag{1}$$

where bosses counts cleared act bosses (0/1/2 by reached floor for non-victory runs, 3 for victories; full mapping in Appendix A). The value is recomputed from outcome, floor, and boss-count fields, not copied from the raw archive `score` field. The 52/3 coefficient calibrates three cleared bosses to 52 points, the approximate mid-Act-3 floor reach. A ±10% perturbation of the coefficient checks score-based qualitative comparisons; win-rate claims do not depend on this score scale.

3.3 Data corpus, release, and harness

The released archive contains 298 completed independent game trajectories spanning fixed-A_0 ablations, cross-backbone probes, and auto-mode ascension runs. Each trajectory records target and reached Ascension, outcome, wall-clock duration, LLM-call counts, condition tag, and the active memory/scaffold setting. The headline fixed-A_0 comparison uses a pre-specified balanced subset of 50 completed games, namely the first ten per condition under the frozen configuration. Other completed trajectories support the cross-backbone and ladder diagnostics rather than entering the fixed-A_0 estimate. Frozen L_4+L_5 stores and per-condition tags are released with the public artifact archive [34, 21].

The game alone is not the full benchmark: a game interface must be paired with a decision protocol. Public Slay the Spire 2 implementations [11, 13, 5, 6] let LLMs act in the game, while the independently evaluated AGI-Eval configurations [1] report no listed victory. Our architecture (§4) supplies the bounded-memory contract that makes the released trajectories a reusable evaluation surface.

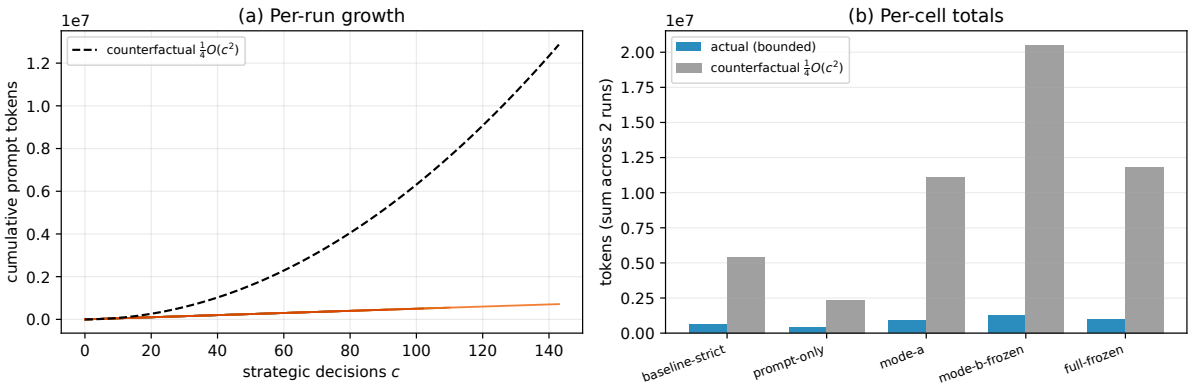

Figure 3 Token audit of the bounded-memory contract on ten fixed-A_0 runs (two per cell); dashed line is a transcript-appending counterfactual at $\frac{1}{4}$ of naive $O(c^2)$ growth (median tokens/call $\times c(c{+}1)/8$, a moderate prompt-caching discount). The audit illustrates context-growth mechanics under each contract; at two runs per cell it is not a win-rate comparison.

4 Architecture: Per-Decision Typed Retrieval

This section specifies what the LLM is allowed to see when it makes a move. The agent never appends the raw message turns from earlier decisions. Instead, it rebuilds each decision prompt from five typed knowledge layers (Figure 2b). Any information that survives across decisions must first be written into a bounded store; in our experiments, postrun extraction and skill discovery can write back only to L_4/L_5.

The contract gives the resource four evaluation handles that a raw prompt-history setup usually hides: horizon growth is capped by slot budgets; retrieved evidence is labeled by layer; L_4 and L_5 can be toggled without rewriting the whole prompt; and runs, stores, prompts, and scripts carry condition tags for reuse.

4.1 Per-decision compositional context

In ReAct/Reflexion-style agents [47, 28], the model may see a growing log of earlier states, tool calls, and self-critiques. AgenticSTS uses a different interface: for decision d at state s_d, the engine retrieves from $L_1, \ldots, L_5$ and composes a fresh user message

$$u_d = \pi\big(L_1, L_2(s_d), L_3(s_d), L_4(s_d), L_5(s_d)\big), \tag{2}$$

sent to the LLM as $\langle \text{sys}, u_d \rangle$. The design still allows bounded typed summaries, a per-run Strategic Thread, and same-decision repair retries. What it disallows is an unbounded transcript that grows because the run has been long.

Bounded context. With capped top-k retrieval and capped item sizes, the configured prompt size is $O(|\text{sys}| + s_{\text{thread}} + \sum_i k_i \cdot s_i)$. The raw cross-decision transcript therefore does not scale with the number of decisions. A transcript interface has worst-case $\Omega(d \cdot \bar{s})$ growth for d decisions, raising per-decision token cost as a run lengthens [22]. Figure 3a–b reports a per-cell linearity audit ($N{=}2$ runs per cell, 10 runs total) of the released fixed-A_0 runs.

Ablatable layers. Because context reaches the model through named slots, we can switch prompt strictness, rule retrieval, episodes, and strategic skills on or off independently. This is the main experimental advantage of the contract: the fixed-A_0 matrix can ask which layer changes behavior, not only whether a larger prompt helps.

4.2 Five typed knowledge layers

The compose operator uses five substrates, separated by mutability and role. L_1 *operator prompts* contain immutable role and protocol templates for each state type. L_2 *state-typed prompts* provide immutable schemas

for combat, deckbuilding, map, event, and intermission decisions, including legal action formats. L_3 *game knowledge* stores enumerable rule data—cards, relics, events, enemies, and intents—refreshed by patch. L_4 *episodic memory* stores postrun summaries (character $\times$ ascension $\times$ act $\times$ enemy class) — *case-based recall*. L_5 *skill library* stores triggered strategic guides distilled from logs — *general scenario-class tactics* indexed by trigger conditions for retrieval across recurring state classes. Each L_5 guide has an explicit trigger, a prose policy, and a four-level write gate.

Raw Slay the Spire 2 logs are not used as similarity RAG. In this game, nearby-looking states can have very different strategic meanings because of card order, relic combinations, and route history. The agent therefore retrieves summaries and triggered guides rather than nearest-neighbor log snippets.

4.3 Routing and combat truncation

A dispatcher routes decisions to four model tiers: *fast* for trivial combat plans, *strategic* for ordinary decisions, *analysis* for postrun memory extraction, and *evolution* for skill distillation. Four static system prompts are cacheable; per-run state is placed in the user message. Combat is the only decision type with a local conversation object, and that object emits at most three messages per round: `combat_start`, `ok`, and the latest user state. Earlier rounds are summarized through the typed state rather than appended. Together with fast-tier routing and mechanical handlers, this yields a median of 67 strategic LLM calls per run instead of one call for every in-game action.

4.4 L_4 episodic memory: role and ablation

L_4 is the episodic layer. It stores postrun summaries for later retrieval and can also feed online skill evolution. In the fixed-A_0 matrix, `full-frozen` (with L_4) and `mode-a` (without L_4) both win 6/10 games. Thus the balanced A_0 comparison points to L_5 as the layer associated with the headline lift. L_4 remains part of the longer-horizon substrate used in the auto-mode streams, which attempt A_6–A_8.

4.5 Skill discovery: distillation, write gate, and Mode B

L_5 is populated in two ways. **Mistake-driven discovery** (*self-evolve*) reads combat losses relative to per-enemy baselines, runs a pre-write A/B check (B=3 resample, strict 2/3 plus zero-harmful), and then applies a four-level write gate: cosine, Jaccard, LLM judge, and optional reap. Most candidates are rejected or merged rather than added as new skills. **Stub-template-filled authoring** (*Mode B*) fills five character-parametric templates for combat, boss, deckbuilding, map, and intermission decisions, under namespace isolation, a library lock, and warn-only validators. Mode B reaches the same 6/10 fixed-A_0 point estimate as the human-authored `mode-a` seed library (§6.4), so the evaluation can separate the existence of a skill layer from the prose source used to populate it.

5 Experimental Methodology

The evaluation is organized around three empirical questions. At a single difficulty, which prompt and memory layers matter? If a frozen L_4+L_5 stack is moved to another backbone, does it still help? When postrun writing is allowed, how far does the agent climb on the Ascension ladder?

5.1 The 5-condition decomposition at fixed A_0

The fixed-A_0 study is a five-cell decomposition, not a full factorial grid (Figure 4). `baseline-strict` uses the baseline prompt with no memory, no skills, no strategic thread, and no combat-conversation wrapper; it also applies the strict knowledge/hint filter. `prompt-only` keeps the full prompt and conversation helpers but disables L_4 and L_5. `mode-a` adds human-authored L_5 seed bodies. `mode-b-frozen` replaces those bodies with stub-template-filled L_5 bodies. `full-frozen` adds the frozen L_4 episodic store to Mode A. In all five cells, postrun and evolution writes are disabled, anchoring the active stores at SHA `1888a62`.

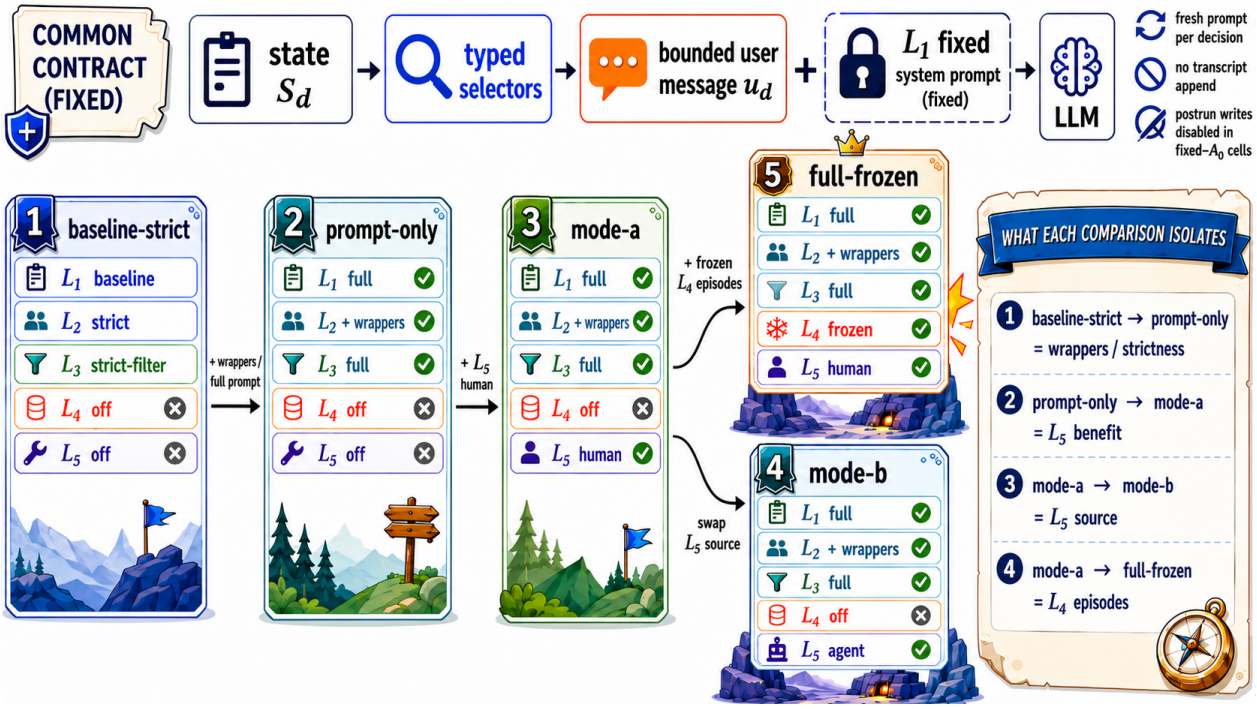

Figure 4 Fixed-A_0 ablation surface under the bounded-memory contract: five cells share a common contract and adjacent bars are organized by a named ablation axis. Not every neighboring pair is a single-mechanism isolation—e.g. baseline-strict to prompt-only changes the prompt-package/strictness group rather than one isolated mechanism. Frozen stores at SHA `1888a62`.

5.2 Cross-backbone probe and auto-mode ladder

The frozen L_4+L_5 stack was derived from Gemini 3.1 Pro trajectories. To test how backbone-specific that stack is, we run `baseline-strict` and `full-frozen` at A_0 on three backbones: Qwen 3.6 27B, DeepSeek V4 Pro, and Gemini 3.1 Pro. The Qwen and DeepSeek probe adds N=5 completed games per backbone-cell; the Gemini rows reuse the corresponding fixed-A_0 cells as anchors. We report this probe separately from the headline ablation; Appendix A lists the denominator rules.

The auto-mode ladder uses a different protocol. After a victory at A_n, the next run attempts A_{n+1}; after a defeat, it retries A_n. This stream measures the observed climb endpoint, whereas the fixed-A_0 matrix measures reliability at one difficulty.

5.3 Statistical protocol

Cell-level win rates use Wilson 95% confidence intervals [43]. Continuous scores (Eq. 1) use 5,000-bootstrap 95% intervals [9]. The descriptive pooled scaffolded row uses an exact Clopper–Pearson interval and is labeled as such. For the headline fixed-A_0 table, we take the first ten completed games per condition by start time, giving a balanced 50-game comparison. For context, recent LLM-agent game evaluations typically report 3–25 episodes per condition—e.g., 3 trials per cell in Voyager [36] and 10–25 seeds per task in BALROG [26]—so the present balanced 5×10 subset sits at or above the comparable range while keeping completed-run denominators interpretable. Completed games are the denominator throughout. Additional completed games in the archive remain in their diagnostic streams and are not pooled into the fixed-A_0 estimate. Score-based qualitative comparisons were also checked after perturbing the 52/3 coefficient by ±10%; win-rate intervals are unaffected.

Table 2 Fixed-A_0 ablation (N=10/cell). Wilson 95% CIs $[11, 60]/[17, 69]/[31, 83]$ for 3/10/4/10/6/10.

Cell	L_5	L_4	Win	Score
No scaffold	–	–	3/10	70.4
Prompt only	–	–	4/10	69.6
Hand skills	A	–	**6/10**	**85.5**
Template skills	B	–	**6/10**	83.3
Skills+episodes	A	✓	**6/10**	82.1

5.4 Reproducibility and release contents

The public release includes the completed-run archive, condition tags, analysis scripts, frozen memory/skill snapshots, and representative prompt records needed to recompute the reported win rates [34, 21]. The same archive defines the natural next matched experiment: an accumulating-context condition implemented in the same codebase with the same run protocol and scoring scripts. Scope limits that affect interpretation are noted where they arise and collected in the Limitations section.

6 Results

6.1 Public difficulty calibration

External rows calibrate difficulty, not causal attribution. The May 2026 AGI-Eval snapshot reports zero listed A_0 victories across five frontier-model configurations [1] (max defeat floor 33), and Mega Crit reports a player-side A_0 win rate of 16% across 240M community runs [23]. Under our own harness, `baseline-strict` wins 3/10 runs (Wilson 95% CI $[10.8, 60.3]$; mean score 70.4), placing the task in a hard but non-saturated regime. Public rows use different interfaces, prompt budgets, and decoding setups, so they are not matched baselines; the within-harness ablation in §6.2 establishes the role of typed retrieval.

6.2 Within-harness ablation

Table 2 reports the balanced fixed-A_0 subset: ten completed games per condition, the active L_5/L_4 layers, and the mean derived score. The comparison isolates prompt strictness, triggered skills, and episodic memory inside one codebase.

The largest separation is between no-scaffold and skill-scaffolded rows. The two no-scaffold cells win 3/10 and 4/10 games; all three L_5 cells win 6/10. Writing the layer-attributable difference as

$$\Delta_{L_\ell} = \widehat{p}_{\text{with-}\ell} - \widehat{p}_{\text{without-}\ell}, \tag{3}$$

Table 2 gives Δ_{prompt}=+1/10 (strictness, wrappers) and Δ_{L_5}=+2/10 at the same prompt setup. *At N=10 this difference is not statistically significant*: a Fisher exact test on 3/10 vs. 6/10 gives $p \approx 0.37$, and pooling all scaffolded vs. unscaffolded cells (18/30 vs. 7/20) gives $p \approx 0.148$; the Wilson 95% CIs [43] overlap. We therefore read L_5 as the layer with the largest observed difference in the balanced matrix — a directional result, not a fine ranking among the three scaffolded variants and not a significance claim. Establishing whether the bounded contract itself outperforms a matched accumulating-context design would require the controlled comparison we leave to future work (Limitations).

6.3 Auto-mode ascension ladder: endpoint evidence

Runs with postrun-active memory attempt A_6–A_8, while no-postrun streams stop at A_2–A_4 (Figure 5). The ladder therefore complements the fixed-A_0 matrix: one stream isolates component lift at a fixed difficulty, and the other shows the highest difficulty reached when stores can be updated after runs.

6.4 Template skills, transfer, and episodes

Mode B without hand-authored skill prose. `mode-b-frozen` matches the hand-authored `mode-a` 60% A_0 win estimate (score-diff CI $[-18.6, +24.8]$): template-filled skills are competitive with the seed library inside the

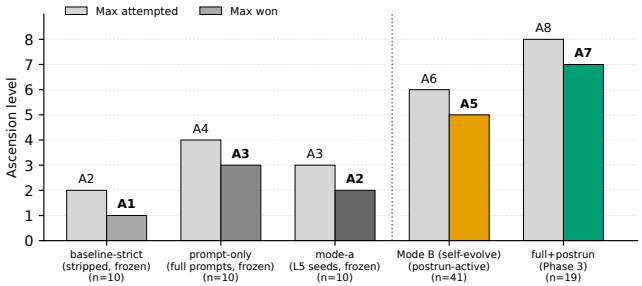

Figure 5 Auto-mode ascension ladder: per-stream highest attempted ascension (endpoint, not win-rate).

Table 3 Cross-backbone transfer of the Gemini-trained $L_4{+}L_5$ stack ($N{=}5$/cell, Gemini $N{=}10$). Full-frozen 95% score CIs $[13.8, 41.9]/[21.7, 45.9]/[63.1, 96.5]$ for Qwen/DeepSeek/Gemini; Qwen and DeepSeek wins $= 0/5$ in both columns, so $\Delta\%$ is a score-only signal. †Gemini floor-48 endpoints include victories.

Backbone	Wins	Score	$\Delta\%$	Floor
Qwen 3.6-27B	$0/5 \rightarrow 0/5$	$14.6 \rightarrow$ **26.9**	$+84.5$	$17 \rightarrow$ **33**
DeepSeek V4-Pro	$0/5 \rightarrow 0/5$	$41.3 \rightarrow 33.8$	-18.1	$37 \rightarrow 33$
Gemini 3.1-Pro	$3/10 \rightarrow$ **6/10**	$70.4 \rightarrow$ **82.1**	$+16.6$	$48 \rightarrow 48^{\dagger}$

same skill interface.

Frozen skills are backbone-sensitive. Table 3 shows the same Gemini-trained `full-frozen` $L_4{+}L_5$ stack transferred to two non-training backbones. The Gemini-trained stack lifts Qwen's mean score ($+84.5\%$) but reduces DeepSeek's (-18.1%); Qwen and DeepSeek wins remain $0/5$. Transfer is therefore an empirical property of the stack, not a premise.

L_4 at A_0 is saturated. `mode-a` (no L_4) and `full-frozen` (with L_4) produce the same win point estimate (score-diff CI $[-21.7, +14.9]$); L_4 still serves the longer-horizon substrate in the ladder streams (§6.3).

7 Comparison with Open-Source Accumulating-Context Agents

The submitted version compared against external calibration anchors only. Here we add a direct, same-testbed *operational* comparison against the two open-source StS2 agents that could be run faithfully end-to-end: **STS2MCP** [11] and **CharTyr** [6]. Both follow the dominant agent-loop design our contract argues against: a single accumulating chat transcript, re-sent (and grown) on every decision. We stress at the outset that this is a comparison of *shipped systems*, not a controlled ablation of the memory contract: the competitors differ from our agent in game patch, routing, thinking effort, decision batching, and prompt cadence as well as in the contract, so the gaps below characterize the current state of practice rather than isolating boundedness as the cause.

Faithful replication. Each competitor runs its *author-intended* configuration: the author's own mod build (minimal load-compatibility patches only; zero agent-logic changes), the author's own MCP server, and the author's own skill/strategy documents as the system prompt, driven through their tool interface. A leak audit over the captured requests confirms no content from our project enters their context. Exactly one mod is loaded at a time, and every game is a fresh Silent A_0 run on the same machine and the same v0.103.x line of the game: our cells ran on v0.103.1; a minor game patch (v0.103.3, 2026-05-30) landed between the two batches, so competitor runs used v0.103.3 (their mods compile against the v0.103.1-pinned reference and load cleanly on v0.103.3; our own stack re-verified on v0.103.3). All strategic decisions for every agent run on `gemini-3.1-pro-preview`; our agent additionally routes trivial decisions to a flash-lite fast tier and sets explicit thinking effort, while competitors run at the provider default with no thinking parameter (their intended setup). The denominator is completed games — harness failures are re-run, never counted as losses — and every run ended in a natural in-game terminal, none at the decision cap. All raw prompts, responses, token usage, and per-step game state are released (released with the data archive).

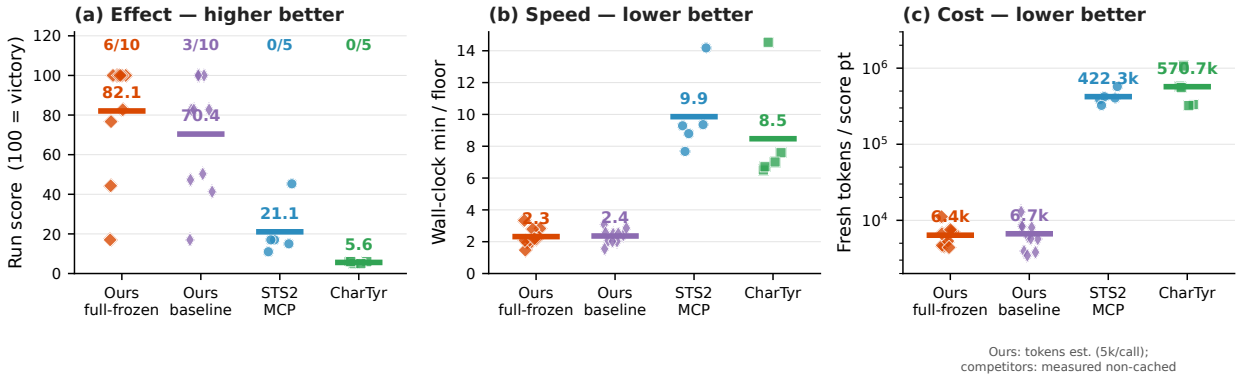

Figure 6 Competitor comparison at A_0 (The Silent), per run; horizontal bars are per-cell means. **(a) Effect**: run score ($s = 100$ if victory else floor $+ (52/3) \cdot$ bosses), with win counts above each column. **(b) Speed**: wall-clock minutes per floor reached (96% of competitor wall-clock is provider-reported LLM latency; fixed inter-action delays slightly favor competitors, 0.5s vs. our 0.6s; our durations exclude postrun). **(c) Cost**: fresh (non-cached) LLM tokens per score point (log). Competitor usage is exact provider-reported tokens with cache hits removed (90% / 82% of their prompt tokens were cached); ours follows the paper's Fig. 3 convention (∼5k strategic user-message tokens × `llm_calls`), excluding the cached system prefix, completions, retries, and fast-tier calls. `full-frozen/baseline-strict` are the Table 2 cells (N=10, frozen stores at SHA `1888a62`); competitors N=5. Under the intentionally absurd upper bound pricing *every* recorded action as a full strategic call, our cells move to 55k / 58k tokens per point — the gap remains $\geq 7\times$.

Effect. Figure 6(a) shows run scores ($s = 100$ if victory else floor $+ (52/3) \cdot$ bosses, §3). Both accumulating agents collapse at A_0: STS2MCP wins 0/5 (mean floor 17.6; one Act-1 boss cleared across five runs), CharTyr wins 0/5 (mean floor 5.6; its frequent `invalid_action` interface errors compound into early deaths — a property of the agent under test, faithfully reproduced[1]). Our `full-frozen` cell wins 6/10 (mean score 82.1) and even `baseline-strict` — our harness with the bounded contract but no learned stores — wins 3/10 (70.4), so the gap is not explained by harness quality alone.

Speed and cost. Figure 6(b,c) and Figures 7–8 quantify the operational gap. Per floor reached, the accumulating agents need ∼4× the wall clock (9.9 / 8.5 vs. 2.3 minutes; 96% of their wall-clock is provider-reported LLM latency, so the gap is not harness pacing). Per score point, they spend 66–90× more *fresh* (non-cached) LLM tokens; under raw ingested context the multiplier exceeds 450×, and even pricing every recorded action of ours as a full strategic call (an intentionally absurd upper bound) leaves $\geq 7\times$. Part of this gap is decision batching — one strategic call drives multiple actions in our agent, while the competitors call the LLM once per action by design — which we report as a property of the memory-architecture package rather than of the backbone. Figure 7 shows the mechanism: their per-call prompt grows from ∼9k to 500k tokens within a single run (trimming caps message *count*, but late-game states grow each message), while the bounded contract holds the strategic user message flat at a ∼5k median.

What this comparison does and does not show. It does not show that accumulating context can never win StS2 — both competitors are community projects, not tuned baselines, and CharTyr's losses are partly interface errors. It does show that the two publicly available transcript-accumulating agents, run faithfully on the same backbone, game line, character, and ascension, with the denominator rules of our own evaluation, fall far below even our no-store bounded baseline while consuming one to two orders of magnitude more tokens per point of progress. A matched same-codebase accumulating-context cell would isolate the contract variable itself; we leave that controlled comparison to follow-up work, and this release is organized to support it. The present section establishes the external state of practice under disclosed operational differences.

[1] CharTyr outcomes for runs 2–5 are inferred from terminal floors 5–6 plus the absence of any victory flag anywhere in the captures; run 1's defeat is confirmed by its captured structured `game_over.is_victory=false`. All captures are released for audit.

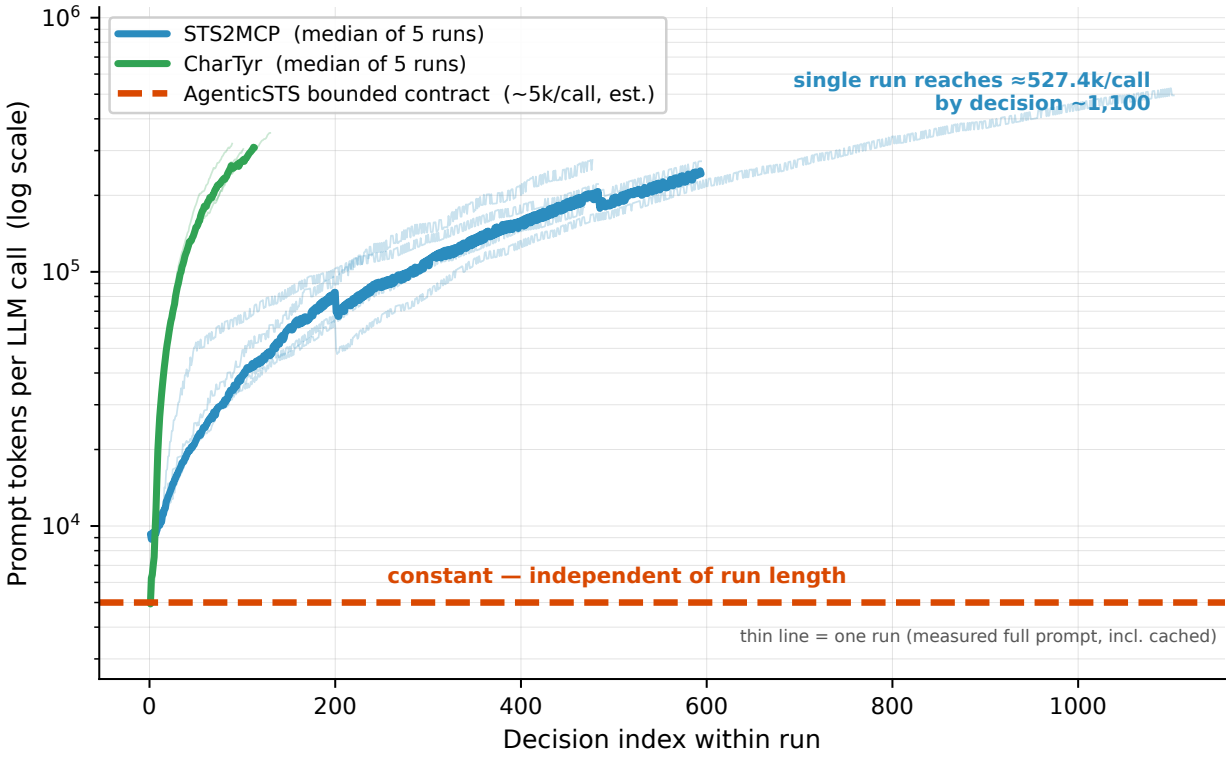

Figure 7 The mechanism: per-call prompt size over a run. Thin lines are individual runs (measured full prompts, including cached tokens — caching changes billing, not what the model attends to); bold lines are per-competitor medians. The worst single STS2MCP run reaches ∼500k tokens per call by decision ∼1100. The dashed line is our bounded contract's strategic user-message median (∼5k, estimate; constant cached system prefix excluded; x-extent not comparable — our runs make ∼100 strategic calls).

8 Discussion

Interpretation and scope. The main lesson is that the memory interface can be made into an object of evaluation rather than left as a prompting convention. In a closed-rule game with text-readable state, the bounded typed contract supports fixed-A_0 wins, locates the largest within-harness difference at the L_5 skill layer (directional at our sample size), and keeps fixed-difficulty performance separate from ladder endpoints. The release is meant to make the next comparison easier: an accumulating-context variant can be added in the same codebase, with the same condition tags, frozen stores, prompt records, and scoring scripts, rather than inferred from public runs that use different harnesses.

Implications. Two observations suggest broader applicability. First, separating memory into typed slots makes attribution tractable: gains can be traced to a specific layer rather than to "more context." Second, the bounded contract decouples interface design from accumulating state, making the same evaluation surface portable to non-game agentic tasks with comparable closed-rule structure.

Implications for loop engineering. The bounded contract is a concrete, measurable design point for the memory stage of closed-rule, turn-based agent loops: per-decision typed retrieval keeps the online context bounded regardless of run length, typed stores make memory updates auditable, and postrun writes expose learning as explicit artifacts rather than opaque transcript growth. Whether this pattern is the right default for open-ended production loops is untested here; what we provide is a reproducible testbed and archive for measuring a memory-layer change under fixed game, denominators, and scoring—an empirical complement to the largely qualitative loop-engineering guidance now common in practice [2, 15].

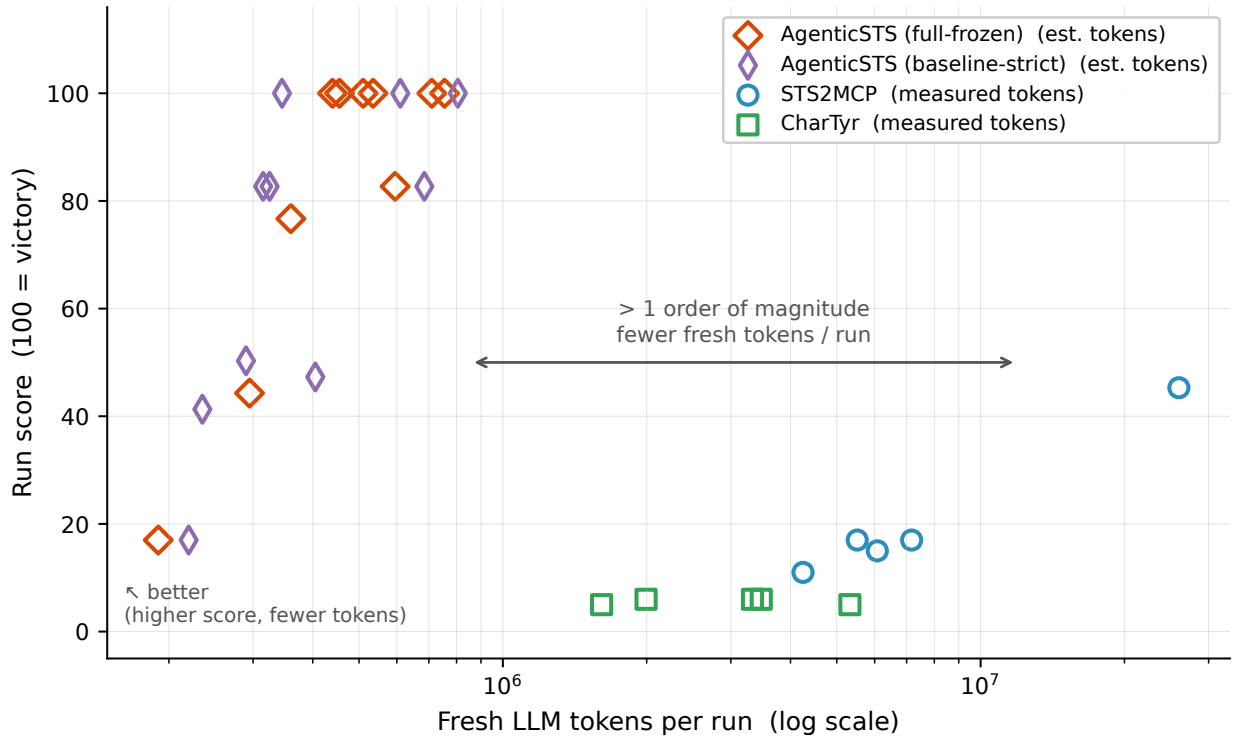

Figure 8 Cost–effect frontier: run score vs. fresh LLM tokens per run (log). Hollow diamonds: our cells (tokens estimated, 5k/call convention); circles/squares: measured competitor usage. The empty band between the clusters spans more than an order of magnitude.

9 Conclusion

We present a long-horizon LLM-agent resource where memory is bounded, typed, and ablatable at decision time. In the fixed-A_0 matrix the largest observed difference appears when triggered L_5 skills are enabled, though at this sample size it is directional rather than statistically decisive; the 298-trajectory archive supplies logs, frozen stores, prompts, and scripts. By keeping game, denominators, stores, and prompts aligned, the release lets future work ask whether a new memory interface changes decisions rather than whether it benefited from a different evaluation scaffold. Our evidence supports a narrower conclusion: explicit, typed memory contracts make long-horizon agent behavior easier to audit, reproduce, and ablate. Whether a bounded contract outperforms a matched accumulating-context design remains an open question for a controlled follow-up.

Limitations

Sample size and statistical inference. The headline fixed-A_0 result uses a balanced 50-game subset, which is at or above the per-condition sample size typical for recent LLM-agent game benchmarks [36, 26, 14]. The cross-backbone and ladder streams are smaller diagnostic streams, reported separately rather than pooled into the headline table. Wilson intervals at this sample size are read at the scaffolded-versus-unscaffolded level; finer-grained equivalence tests among scaffolded variants and smooth backbone-transfer curves are natural extensions that the released archive supports.

Same-codebase accumulating-context variant. The release is organized so that an accumulating-context cell sharing our codebase, condition tags, scoring scripts, and frozen stores can be added as one further row in the same matrix. This is the cleanest direct comparison to the bounded contract, and the artifact provides the run protocol, prompt records, and analysis scripts needed to run it.

Single character and game-version coverage. The headline runs target one playable character, Silent, chosen to keep the typed substrate (L_3 enumerable game knowledge, L_4+L_5 stores) self-consistent in the present submission. Released trajectories carry game-version tags so that version-stratified re-analysis is possible from the archive, and cross-character runs follow the same harness once $L_3/L_4/L_5$ are repopulated for the new character.

External player and ecosystem references. Mega Crit, Spiracle, and AGI-Eval enter the paper only as difficulty and ecosystem context, with cached snapshots released alongside the artifact. Matched human inference would require a separately designed user study with controlled denominators, which is outside the present resource.

Architectural scope. The evaluation is training-free and single-game. The bounded contract is tuned for *turn-based decision* settings like Slay the Spire 2; continuous or streaming control loops, visual input, multi-agent play, online human correction, model-internal fine-tuning, and cross-game transfer are deliberate non-targets of this release. Stub templates and expert seed skills are author-curated; Mode B measures within-interface template filling, which we report as one operating point rather than as fully autonomous skill invention.

References

[1] AGI-Eval Community. Slay the Spire 2 Becomes DeepSeek V4's Mirror: Sounds Reasonable, Plays Terribly. Community blog post (Chinese), https://deepseek.csdn.net/6a01b6b80a2f6a37c5a944ed.html; video https://www.youtube.com/watch?v=0v94pZmif9Y, May 2026. Non peer-reviewed; cited as cross-harness calibration only; accessed 2026-05-13.

[2] Anthropic. Effective context engineering for AI agents. Anthropic Engineering blog, https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents, 2025. Non peer-reviewed industry article; accessed 2026-06-17.

[3] B. Bateni and J. Whitehead. Language-Driven Play: Large Language Models as Game-Playing Agents in Slay the Spire. In *Proceedings of the 19th International Conference on the Foundations of Digital Games*, 2024.

[4] T. Bertram. UrzaGPT: LoRA-Tuned Large Language Models for Card Selection in Collectible Card Games, 2025.

[5] biolbe1230. AI-Spire: LLM plays Slay the Spire 2 through prompt engineering. GitHub repository, https://github.com/biolbe1230/ai-spire, 2026. commit b0a40997; accessed 2026-05-13.

[6] CharTyr. STS2-Agent: MCP server for Slay the Spire 2. GitHub repository, https://github.com/CharTyr/STS2-Agent, 2026. commit 2617fb19; accessed 2026-05-13.

[7] P. Chhikara, D. Khant, S. Aryan, T. Singh, and D. Yadav. Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory, 2025.

[8] J. Du, J. Wu, Y. Chen, Y. Hu, B. Li, and J. T. Zhou. Rethinking Agent Design: From Top-Down Workflows to Bottom-Up Skill Evolution, 2025.

[9] B. Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

[10] S. Forouzandeh, W. Peng, P. Moradi, X. Yu, and M. Jalili. Learning Hierarchical Procedural Memory for LLM Agents through Bayesian Selection and Contrastive Refinement, 2025.

[11] Gennadiyev. STS2MCP: Full agentic runs for Slay the Spire 2. GitHub repository, https://github.com/Gennadiyev/STS2MCP, 2026. commit 2fb53908; accessed 2026-05-13.

[12] D. Hafner. Benchmarking the spectrum of agent capabilities. In *International Conference on Learning Representations (ICLR)*, 2022.

[13] hiKareeem. ClaudePlaysTheSpire (HermesBridge): Claude and friends play Slay the Spire II. GitHub repository, https://github.com/hiKareeem/ClaudePlaysTheSpire, 2026. commit 38202d7f; accessed 2026-05-13.

[14] L. Hu, M. Huo, Y. Zhang, H. Yu, E. P. Xing, I. Stoica, T. Rosing, H. Jin, and H. Zhang. lmgame-Bench: How Good are LLMs at Playing Games?, 2025.

[15] Y. Hu, S. Liu, Y. Yue, et al. Memory in the age of AI agents. *arXiv preprint arXiv:2512.13564*, 2025.

[16] Y. Jiang, D. Li, H. Deng, B. Ma, X. Wang, Q. Wang, and G. Yu. SoK: Agentic Skills – Beyond Tool Use in LLM Agents, 2026.

[17] J. Kang, M. Ji, Z. Zhao, and T. Bai. Memory OS of AI Agent, 2025.

[18] T. Kwa, B. West, J. Becker, A. Deng, K. Garcia, M. Hasin, S. Jawhar, M. Kinniment, N. Rush, S. Von Arx, R. Bloom, T. Broadley, H. Du, B. Goodrich, N. Jurkovic, L. H. Miles, S. Nix, T. Lin, N. Parikh, D. Rein, L. J. K. Sato, H. Wijk, D. M. Ziegler, E. Barnes, and L. Chan. Measuring AI Ability to Complete Long Software Tasks, 2025.

[19] H. Küttler, N. Nardelli, A. H. Miller, R. Raileanu, M. Selvatici, E. Grefenstette, and T. Rocktäschel. The NetHack Learning Environment. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[20] X. Li, W. Chen, Y. Liu, S. Zheng, X. Chen, Y. He, Y. Li, B. You, H. Shen, J. Sun, S. Wang, B. Li, Q. Zeng, D. Wang, X. Zhao, Y. Wang, R. B. Chaim, Z. Di, Y. Gao, J. He, Y. He, L. Jing, L. Kong, X. Lan, J. Li, S. Li, Y. Li, Y. Lin, X. Liu, X. Liu, H. Lyu, Z. Ma, B. Wang, R. Wang, T. Wang, W. Ye, Y. Zhang, H. Xing, Y. Xue, S. Dillmann, and H.-c. Lee. SkillsBench: Benchmarking How Well Agent Skills Work Across Diverse Tasks, 2026.

[21] C. Liu, L. Zhang, X. Xu, W. Guo, and Y. Liu. Towards the versioning of llm-agent-based software. In *Companion Proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering (FSE Companion '25)*, pages 1619–1622, 2025. Ideas, Visions and Reflections track (4-page paper).

[22] E. Lumer, F. Nizar, A. Jangiti, K. Frank, A. Gulati, M. Phadate, and V. K. Subbiah. Don't Break the Cache: An Evaluation of Prompt Caching for Long-Horizon Agentic Tasks, 2026.

[23] Mega Crit. The Neowsletter - May 2026. Steam Community announcement, https://store.steampowered.com/news/app/2868840/view/701016542742053855, May 2026. Developer newsletter for Slay the Spire 2; non peer-reviewed; accessed 2026-05-23.

[24] S. Ouyang, J. Yan, Y. Chen, R. Han, Z. Wang, B. D. Mishra, R. Meng, C.-L. Li, Y. Jiao, K. Zha, M. Shen, V. Tirumalashetty, G. Lee, J. Han, T. Pfister, and C.-Y. Lee. SkillOS: Learning Skill Curation for Self-Evolving Agents, 2026.

[25] C. Packer, S. Wooders, K. Lin, V. Fang, S. G. Patil, I. Stoica, and J. E. Gonzalez. MemGPT: Towards LLMs as Operating Systems, 2024.

[26] D. Paglieri, B. Cupial, S. Coward, U. Piterbarg, M. Wolczyk, A. Khan, E. Pignatelli, L. Kucinski, L. Pinto, R. Fergus, J. N. Foerster, J. Parker-Holder, and T. Rocktaschel. BALROG: Benchmarking Agentic LLM and VLM Reasoning On Games, 2024.

[27] J. S. Park, J. C. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein. Generative Agents: Interactive Simulacra of Human Behavior, 2023. ACM Symposium on User Interface Software and Technology (UIST).

[28] N. Shinn, F. Cassano, A. Gopinath, K. R. Narasimhan, and S. Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[29] A. Sinha, A. Arun, S. Goel, S. Staab, and J. Geiping. The Illusion of Diminishing Returns: Measuring Long Horizon Execution in LLMs, 2025.

[30] Spire Codex. Spire Codex: Slay the Spire 2 Database. Community database, https://spire-codex.com/, 2026. Public game-data database and REST API; non peer-reviewed; accessed 2026-05-23.

[31] STS2 Community Stats. Slay the Spire 2 Community Stats. Community statistics website, https://www.sts2.fun/, 2026. Community-uploaded survival-by-floor and ascension statistics; non peer-reviewed; accessed 2026-05-25.

[32] T. R. Sumers, S. Yao, K. Narasimhan, and T. L. Griffiths. Cognitive Architectures for Language Agents, 2024. Transactions on Machine Learning Research.

[33] W. Tang, Y. Zhou, E. Xu, K. Cheng, M. Li, and L. Xiao. DSGBench: A Diverse Strategic Game Benchmark for Evaluating LLM-based Agents in Complex Decision-Making Environments, 2025.

[34] S. Tripathi, D. Alkhulaifat, F. X. Doo, P. Rajpurkar, R. McBeth, D. Daye, and T. S. Cook. Development, Evaluation, and Assessment of Large Language Models (DEAL) Checklist: A Technical Report. *NEJM AI*, 2(6), May 2025. NEJM AI uses article-number citation (AIp2401106) rather than traditional page numbers; online publication 2025-05-22.

[35] B. Wang, K. McKeown, and R. Ying. DYSTIL: Dynamic Strategy Induction with Large Language Models for Reinforcement Learning, 2025.

[36] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar. Voyager: An Open-Ended Embodied Agent with Large Language Models. *Transactions on Machine Learning Research*, 2024. Originally released as arXiv:2305.16291.

[37] J. Wang, Q. Yan, Y. Wang, Y. Tian, S. S. Mishra, Z. Xu, M. Gandhi, P. Xu, and L. L. Cheong. Reinforcement Learning for Self-Improving Agent with Skill Library, 2025.

[38] W. Wang, F. Bie, J. Chen, D. Zhang, S. Huang, E. Kharlamov, and J. Tang. Can Large Language Models Master Complex Card Games?, 2025.

[39] X. J. Wang, H. Bai, Y. Sun, H. Wang, S. Zhang, W. Hu, M. Schroder, B. Mutlu, D. Song, and R. D. Nowak. The Long-Horizon Task Mirage? Diagnosing Where and Why Agentic Systems Break, 2026.

[40] Z. Wang, K. Wang, Q. Wang, P. Zhang, L. Li, Z. Yang, X. Jin, K. Yu, M. N. Nguyen, L. Liu, E. Gottlieb, Y. Lu, K. Cho, J. Wu, L. Fei-Fei, L. Wang, Y. Choi, and M. Li. RAGEN: Understanding Self-Evolution in LLM Agents via Multi-Turn Reinforcement Learning, 2025.

[41] Z. Wang, F. Wu, H. Wang, X. Tang, B. Li, Z. Yin, Y. Ma, Y. Li, W. Sun, X. Chen, and Y. Ye. Why Reasoning Fails to Plan: A Planning-Centric Analysis of Long-Horizon Decision Making in LLM Agents, 2026.

[42] Z. Z. Wang, J. Mao, D. Fried, and G. Neubig. Agent Workflow Memory, 2024.

[43] E. B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927.

[44] Z. Wu, H. Zhang, F. Lin, W. Xu, X. Xu, Y. Chen, H. P. Zou, S. Chen, W. Zhang, X. Liu, P. S. Yu, and H. Wang. GAM: Hierarchical Graph-based Agentic Memory for LLM Agents, 2026.

[45] C. Xiao, Y. Zhang, X. Huang, Q. Huang, J. Chen, and P. Sun. Mastering Strategy Card Game (Hearthstone) with Improved Techniques. In *IEEE Conference on Games (CoG)*, 2023.

[46] W. Xu, Z. Liang, K. Mei, H. Gao, J. Tan, and Y. Zhang. A-MEM: Agentic Memory for LLM Agents, 2025.

[47] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations (ICLR)*, 2023.

[48] K. Zhang, D. Liu, Q. Zhao, J. Hou, X. Zhang, Q. Xie, M. Liu, and Y. Li. GameVerse: Can Vision-Language Models Learn from Video-based Reflection?, 2026.

[49] A. Zhao, D. Huang, Q. Xu, M. Lin, Y.-J. Liu, and G. Huang. ExpeL: LLM Agents Are Experiential Learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19632–19642, 2024.

[50] B. Zheng, M. Y. Fatemi, X. Jin, Z. Z. Wang, A. Gandhi, Y. Song, Y. Gu, J. Srinivasa, G. Liu, G. Neubig, and Y. Su. SkillWeaver: Web Agents can Self-Improve by Discovering and Honing Skills, 2025.

[51] H. Zhou, S. Guo, A. Liu, Z. Yu, Z. Gong, B. Zhao, Z. Chen, M. Zhang, Y. Chen, J. Li, R. Yang, Q. Liu, X. Yu, J. Zhou, N. Wang, C. Sun, and J. Wang. Memento-Skills: Let Agents Design Agents, 2026.

Appendix

A Evaluation archive and aggregation rule

The archive is organized so that each reported win rate can be recomputed from completed-run records. It includes one record per trajectory, the condition tags used to select cells, frozen L_4+L_5 snapshots for the within-codebase ablation, representative decision-time and postrun prompt records, and scripts for Wilson 95% win-rate intervals and bootstrap 95% score intervals. A cell's win rate is simply victories divided by completed games in that cell. We keep the fixed-A_0, cross-backbone, ladder, and full-archive streams separate. Table 4 gives the denominator rule for each stream.

Table 4 Denominator map. Only the balanced fixed-A_0 subset enters the headline ablation (Table 2); streams are never pooled.

Stream	N	Metric	Role
Fixed-A_0	50 (5×10)	win+score	headline
Backbone	5/cell	score shift	diagnostic
Ladder	unequal	max A	endpoint
Archive	298	tags+scripts	audit

Score-formula audit. The derived score (Eq. 1) uses bosses $= 0$ when floor < 18, 1 when floor < 34, 2 otherwise, and 3 for victories. Table 5 lets readers reproduce the paper-reported means.

Confidence intervals. For a cell with w wins out of n completed runs, the Wilson 95% interval [43] on the win rate $\hat{p} = w/n$ is

$$\frac{\hat{p} + \frac{z^2}{2n} \pm z\sqrt{\frac{\hat{p}(1-\hat{p})}{n} + \frac{z^2}{4n^2}}}{1 + z^2/n}, \quad z = 1.96. \tag{4}$$

Score intervals use the percentile bootstrap [9]: 5,000 resamples with replacement from the cell's n run-level scores, reporting the empirical $[2.5, 97.5]$ percentiles. The descriptive pooled scaffolded row uses an exact Clopper–Pearson interval and is labeled as such.

Table 5 Per-cell floor and boss-clear counts that reproduce the mean scores reported in Table 2.

Cell	N	Wins	$\overline{\text{floor}}$	$\overline{\text{bosses}}$	$\overline{\text{score}}$
baseline-strict	10	3	39.2	1.80	70.40
prompt-only	10	4	38.4	1.80	69.60
mode-a	10	6	43.9	2.40	85.50
mode-b-frozen	10	6	43.4	2.30	83.27
full-frozen	10	6	42.2	2.30	82.07

B Memory contract: five typed layers

Table 6 gives a compact legend for the five stores used in the paper. The useful distinction is mutability: L_1 and L_2 are fixed, L_3 can be filtered, and L_4/L_5 can be disabled, frozen, or made writable depending on the evidence stream.

Table 6 Five-layer memory contract. Each decision receives typed slices; raw cross-decision transcript is not appended.

Layer	Store	Key	Write	Ablation
L_1	protocol	state	fixed	always
L_2	schema	decision	fixed	always
L_3	rules	entities	refresh	filter
L_4	episodes	char/A/act	postrun	off/on
L_5	skills	trigger	gated	off/A/B

C Concrete prompt and learning examples

Decision-time composed prompt. At decision time, the agent assembles the user prompt in a fixed typed order: fired L_5 skills, L_4 episodes, L_3 game facts, the L_2 typed state prompt, and a schema hint listing valid actions. The order is shared across non-combat decision states. Combat is the only intra-fight stateful container, and its emitted message list per round is bounded to three items: combat start, an acknowledgement, and the latest user state. Earlier rounds are not copied into the transcript; the next round is prompted from a newly composed user message. A shortened floor-7 card-reward excerpt shows the typed layers:

```
## Expert Knowledge (retrieved skills)
**Silent - Draft and Shop Rules** (seed)
Early rewards must solve damage first,
then block, draw, and energy. ...

## Card-Specific Insights
- dodge and roll: delayed block plus Dexterity.
- slice: 0-cost transitional damage.

## Game Knowledge / Card Mechanics
- DodgeAndRoll: Block Next Turn; upgrade +2.
- Slice: Upgrade: Damage +3

## Card Reward
HP: 57/57 (100%) | Gold: 55 | Act: 1 | Floor: 7

## Decision Format (card_reward_action)
Valid actions: choose_reward_card
or choose_reward_alternative
```

Representative learned skill. One learned skill concerns a boss mechanic that exhausts attacks and skills on specific turns (turns 2, 5, 8, and 11). The stored rule instructs the agent to preserve core scaling cards on those turns and to prefer powers or natively exhausting cards. The example shows that L_5 stores state-conditioned tactical rules with Boolean triggers, not generic advice or similarity-retrieved raw logs. Full prompt and learned-skill records are included in the artifact archive.

D External calibration and source boundaries

External Slay the Spire 2 statistics enter the paper as difficulty and ecosystem context. Cached snapshots of each source are released with the artifact archive so that follow-up work can reproduce the calibration rows under identical inputs.

Table 7 External numbers calibrate difficulty and ecosystem context only.

Source	Role	Not used as
AGI-Eval	LLM yardstick	causal baseline
Mega Crit A_0	difficulty anchor	matched human test
Spiracle	community context	population rate

Figure 9 Decision states used in the prompt exhibits: combat planning (left) and shop planning (right).

E Full decision-time prompt exhibits

The exhibits below print the model-facing prompt assembled from the typed substrates. Text inside each box is verbatim system or user prompt content; the box colors and titles are reader annotations for L_1 protocol/schema, L_2 current state and action space, L_3 rules and mechanics, L_4 episodic notes, and L_5 strategic skills.

E.1 Combat decision

The combat exhibit is the longest example: it includes the shared system instruction, setup context, retrieved skills, episodic notes, game rules, and the current round state.

System prompt.

L_1 — Role

```
You are an autonomous Slay the Spire 2 agent playing a complete run. You make every decision to maximize your chance of defeating
the Act 3 boss.
```

L_1 — Output schema

```
## Output Format
Think through your decision, then output your choice in a <decision> tag containing valid JSON.

Example (map):
<decision>
{"action": "choose_map_node", "option_index": 2, "reasoning": "Elite fight for card reward", "strategic_note": "Need AoE damage
for Act 2 hallways"}
</decision>

Example (combat plan):
<decision>
{"plan": [{"type": "card", "card": "Backflip", "target_index": -1}, {"type": "card", "card": "Shiv", "target_index": 0}],
"end_turn": true, "reasoning": "Block first, then chip damage", "note_to_future_self": "Poison at 8, need 2 more turns",
"analysis": {"problem": "Incoming 15 damage", "key_observations": ["Can block 11 with Backflip", "Shiv for 5 chip"],
"candidate_lines": ["Block+Shiv", "All-in damage"], "chosen_line": "Block+Shiv to survive"}}
</decision>

The JSON must match the schema for the current decision type. Every decision requires a "reasoning" field.For combat plans, the
`plan` array is the exact execution order: item 1 happens first, then item 2, etc. Put cards in the same order they should
actually be played.
```

L_3 — Combat rules

```
## Core Combat Rules
- **Turn structure**: Each turn you draw 5 cards, gain 3 energy, and your Block resets to 0. Play cards (costs energy), use
potions (free). At end of turn, all remaining hand cards are discarded.
- **Hand resets every turn**: You get 5 NEW cards each turn from your draw pile. Cards with Retain stay. Cards drawn or created
during THIS turn are part of THIS turn's hand immediately; unless they Retain or explicitly return, they will not stay for next
turn.
- **Hand size limit**: Your hand can hold at most 10 cards. If a draw/add-to-hand effect would exceed 10 cards, excess drawn or
generated cards are discarded or fail to enter your hand. At 10 cards, playing a generator first usually creates only one open
slot: the generator card's own slot.
- **Block resets every turn**: Block only protects you during the upcoming enemy turn. You cannot stockpile it unless a visible
card/power explicitly says Block is retained.
```

```
- **Energy resets to 3**: Unspent energy is wasted. Use ALL your energy each turn.
- **Enemy intents are visible**: Attack (damage value shown), Defend, Buff, Debuff, Status (adds junk cards to your deck).
- **Draw effects resolve immediately**: If you play Acrobatics, Swift Potion, Blade Dance, or any effect that draws/adds cards,
those cards are usable NOW this turn.
- **Draw pile is a forecast, not a reservation**: The draw pile only tells you what you might draw later if nothing changes first.
Any draw/add-to-hand effect this turn changes that forecast immediately.
- **Queue plays for generated cards**: If your `plan` includes a card that ADDS new cards to your hand (Blade Dance / Storm of
Steel / Cloak and Dagger -> Shivs, Infernal Blade -> random Attack, Nightmare -> extra copies, etc.), you MUST also queue the
plays for those generated cards in the same `plan`, placed AFTER the generator. The generated cards exist in hand the instant the
generator resolves -- treat them as part of this turn's available plays. Example: `Cloak and Dagger+` adds 2 Shivs, so a plan
using it should look like `[..., Cloak and Dagger+, Shiv, Shiv, ...]` (2 Shiv plays queued after, as long as energy allows).
Failing to queue them wastes the Shivs and forces a mid-round re-plan.
```

L_5 — HP policy

```
## HP Conservation
- **HP is a run-wide resource**: Every point of HP lost now is HP you won't have for the boss. Even at high HP, cherish every
single point.
- Prefer the defensive line that takes 0 damage over the aggressive line that takes 5. Only trade HP for speed when enemies scale
dangerously (Strength stacking, summons).
- **Potions**: Dying with full potions is the worst outcome. Instant potions (Fire, Block) are safe anytime. Save sustained-buff
potions (Strength, Dexterity, Regen) for boss/elite fights where they provide 3-4x more value.
```

User prompt: setup and retrieved context.

L_2 — Combat start

```
## Combat Start
Encounter type: elite
Act: 3 | Floor: 43
Enemies: Soul Nexus
- Soul Nexus [index=0]: HP 234/234, Block 0 | powers: Weak(1): Weakened creatures deal 25% less damage with Attacks.

Player HP: 66/70 | Block: 0 | Energy: 3/3
Player buffs/debuffs: Dexterity(1): Dexterity improves Block gained from cards.
```

L_2 — Deck

```
## Current Deck (35 cards)
  [Attack] Leading Strike+ x2, Byrd Swoop+(cost=0), Mind Blast(cost=1), Neutralize+[Sown](cost=0), Strike(cost=1)
  [Power] Accuracy(cost=1), Accuracy+(cost=1), Footwork+(cost=1), Infinite Blades(cost=1), Phantom Blades(cost=1), Serpent
  Form(cost=3), Well-Laid Plans(cost=1)
  [Skill] Defend x6, Backflip x3, Leg Sweep x2, Backflip+(cost=1), Blade Dance(cost=1), Blade of Ink(cost=1), Cloak and
  Dagger+(cost=1), Deflect(cost=0), Expertise+(cost=1), Knife Trap(cost=2), Piercing Wail(cost=1), Storm of Steel(cost=1),
  Survivor(cost=1), Up My Sleeve(cost=2)
```

L_2 — Relics

```
## Relics (14)
- Ring of the Snake: At the start of each combat, draw 2 additional cards.
- Large Capsule: Upon pickup, obtain 2 random Relics. Add an additional Strike and Defend to your Deck.
- Parrying Shield: If you end a turn with at least 10 Block, deal 6 damage to a random enemy.
- Blood Vial: At the start of each combat, heal 2 HP.
- Oddly Smooth Stone: Start each combat with 1 Dexterity.
- Letter Opener: Every time you play 3 Skills in a single turn, deal 5 damage to ALL enemies.
- Vambrace: The first time you gain Block from a card each combat, double the amount gained.
- Byrdpip: Upon pickup, gain the card Byrd Swoop. A Byrdpip will accompany you in battles.
- Biiig Hug: Upon pickup, remove 4 cards from your Deck. Whenever you shuffle your Draw Pile, add a Soot into your Draw Pile.
- Red Mask: At the start of each combat, apply 1 Weak to ALL enemies.
- Happy Flower: Every 3 turns, gain Energy.
- Whetstone: Upon pickup, Upgrade 2 random Attacks.
- Music Box: Create an Ethereal copy of the first Attack you play each turn.
- Reptile Trinket: Whenever you use a potion, gain 3 Strength this turn.
```

L_4 — Strategic thread

```
## Strategic Thread
*Your deck-building decisions so far -- fight with this deck's strengths.*

- [card_reward] Committed Shiv plan: scale with Accuracy and Phantom Blades, use Shivs for damage and block with Footwork-scaled
Backflips. Needs to draw through the deck quickly to set up powers.
- [shop] Committed shiv plan: prioritize playing scaling powers like Accuracy and Phantom Blades early, then use high-volume shiv
generators and Expertise to cycle damage; use Leg Sweep and Backflips to survive while setup occurs. Needs consistent shiv
production and block scaling; avoid high-cost non-shiv attacks that clog the hand for Expertise.
- [event] Committed Shiv plan: play powers (Accuracy, Phantom Blades), generate Shivs, and cycle through the deck with Backflips
and Expertise. Keep upgrading key block/draw cards and remove remaining basic Strikes.
- [rest_site] Committed Shiv plan: scale damage with Accuracy and Phantom Blades, using Backflip and Expertise for draw and
defense. Prioritize upgrading key scaling powers to ensure Act 3 boss lethality.
```

L_5 — Retrieved skills

```
## Expert Knowledge (retrieved skills)
*Apply these strategies to the current situation. Deviate only with good reason.*


**Silent - Combat Sequencing** (92%)
Silent wins by converting discard and free cards into tempo; count playable cards, not just energy. Sequence zero-cost and setup
cards first: Pinpoint after skills, Pounce before an expensive skill, and discard Sly cards or other trick cards when that gives a
free play. In the first three hallway fights, default to no-damage lines: if full block is available, take it before racing. From
the fourth hallway fight onward, protect HP aggressively and spend potions when needed. Once defense is stable in a long fight,
prefer poison over fair attacks; keep stacking poison and block instead of taking weak damage trades.
  - Example: Play skills before Pinpoint so it discounts to 0 whenever possible.

**Core Combat Principles** (90%)
Every combat turn: (1) Read enemy intents -- if ALL enemies have non-attack intents (Buff/Debuff/Sleep), it is a FREE offense
window. In multi-enemy fights, some enemies may still attack while others buff -- always account for total incoming damage and
block accordingly. (2) Dead enemies deal zero damage. If your attacks can kill an enemy this turn, prioritize that. (3) Check your
draw pile -- knowing what cards remain tells you what you might draw next turn or from draw effects. Plan your plays considering
both current hand and future draws. (4) Use ALL energy -- unspent energy is permanently wasted. If you have 1 energy left, play a
1-cost card. (5) Multi-enemy: focus fire the most dangerous attacker first. Each kill reduces incoming damage next turn. Use AoE
when 3+ enemies alive. (6) Potions don't cost energy -- use buff potions BEFORE attacks, damage potions to secure kills, defensive
potions vs lethal incoming.
  - Example: Single enemy buffs -> play all attacks. Multi-enemy: one enemy buffs but others attack -> still block for the
  attackers

**Boss and Elite Fight Strategy** (seed)
Boss fights are the make-or-break moments. (1) USE ALL POTIONS -- potions exist for these fights. Dying with unused potions is the
worst outcome. (2) Bosses scale over time -- they gain Strength, add curses. End fights QUICKLY. Front-load damage in the first
3-4 turns. (3) After beating a Boss, HP fully restores between Acts. You CAN play more aggressively -- but first verify you
survive the next enemy turn. Never go all-in if incoming damage would kill you. (4) For specific boss/elite mechanics, use
lookup_enemy or recall_encounter tools to learn their attack patterns before the fight. (5) Consider playing Power cards that
boost your damage or block capacity early in boss fights -- sometimes sacrificing one turn of offense or defense to play a scaling
Power wins the long fight.
```

L_4 — Past experience

```
## Past Experience
*Enemy mechanics and fight structure from past encounters.*
```

L_4 — Past experience

```
### Past Experience
- The boss opens the encounter with moderate attacks (18 to 29 damage) and debuff abilities like Soul Burn and DebuffStrong.
- If the fight extends into later rounds, the boss gains Strength and utilizes massive damage spikes, notably a 43-damage single
hit and a 36-damage (9x4) Maelstrom.
```

L_4 — Combat guide

```
## Combat Guide
*Tactical advice from past encounters.*

- [Guide: Soul Nexus] - **Aggressive Racing:** The cleanest wins bypass the boss's lethal attacks entirely by ending the fight in
exactly 3 rounds. Push extreme damage using Shiv generators and draw/discard loops (Acrobatics, Calculated Gamble,
Reflex/Tactician) to race the boss down.
- **Passive Defense:** Standard block cards cannot keep pace with the boss's scaling. Rely heavily on passive defense engines like
Afterimage combined with 0-cost spam (Shivs) to generate incidental block while maintaining your offensive momentum.
- **Mitigate the Spikes:** If the fight extends past round 3, you will face devastating attacks (a 43-damage swing and a 36-damage
Maelstrom). Save premium damage mitigation like Piercing Wail and Weak strictly for these critical survival turns.
```

L_2 — Enemy patterns

```
## Enemy Patterns
Current round: R1
```

L_5 — Potion strategy

```
## Potion Strategy
Fight: elite | Boss: 8 floors away
- Powdered Demise [INSTANT]: Enemy loses 9 HP at the end of each of its turns.
- Swift Potion [INSTANT]: Draw 3 cards.
```

L_4 — Card notes

```
## Card Notes (from experience)
- neutralize: 0-cost attack. Primarily applies Weak, but also serves as a critical free combo enabler to efficiently scale the hit count of cards like Finisher without spending energy.
- survivor: C-tier starter block. Fine early and with discard synergies, but with Well-Laid Plans do not auto-retain it over rarer swing cards, scaling, or premium defense.
- phantom blades: Power: Your first Shiv played each turn deals bonus damage (+6). ALL Shivs Retain. This is primarily a combo/burst enabler, not just passive scaling. By hoarding 0-cost Shivs in hand over multiple turns, you can unleash massive zero-energy burst to push specific boss phases, bypass alternating immunities (like Test Subject's Nemesis), or secure lethal. High priority in Shiv decks.
- infinite blades: Power: creates 1 Shiv at start of each turn. Slow ramp -- needs 3+ turns to accumulate meaningful value. Scales with Accuracy (+4 per Shiv per Accuracy copy). Compare: Fan of Knives generates more Shivs per turn.
- storm of steel: Discards your ENTIRE hand to generate Shivs. This destroys Retained cards (Well-Laid Plans), Nightmared copies, and defensive tools held for future turns. NEVER play this if you are holding essential mitigation (Apparition, Piercing Wail). Best used to convert unplayable cards, statuses (Slimed), or basic strikes into damage. Excellent synergy with Tingsha or Tough Bandages.
- footwork: Power: permanent +2 Dexterity (upgraded: +3). All Block cards gain +2/+3 Block for rest of combat. Stacks with multiple copies. Unlike Anticipate, this is permanent. Upgrade from +2 to +3 is a significant boost.
- cloak and dagger: 1-cost Skill: 6 Block, generates 1 Shiv (Upgraded: 2). High-tier foundational piece for Shiv engines, scaling defensively with Dexterity (Footwork) and offensively with Accuracy. The upgrade is extremely high priority as it doubles the Shiv output. Keep in mind it plays 2-3 cards total, making it susceptible to Beat of Death and Time Eater restrictions later in runs.
- backflip: 1-cost: block + draw 2. Defends and cycles simultaneously. The draw does not trigger Sly (draw is not discard). Pairs with Dexterity (Footwork) for scaled Block.
- deflect: 0-cost: gain Block for no energy. Value increases with Dexterity (Footwork adds flat Block). Better in decks with more draw -- you see it more often per cycle.
- well-laid plans: A-tier control enabler. Beyond surviving boss cycles, it perfectly enables Sly/discard engines by letting you hold unplayable Sly cards (Haze, Ricochet) until you draw a discard outlet.
- leading strike: 1-cost Attack: Deals damage and adds Shivs to hand. Excellent enabler for Finisher, Fan of Knives, and Exhaust synergies (like Joss Paper) by providing multiple cheap attacks per draw.
- knife trap: Replays EVERY Shiv in your Exhaust pile. Functions as a lethal boss finisher. Base damage is low, so scale it first with Accuracy, Envenom, Vulnerable, or Tracking before unleashing the swarm.
- serpent form: Rare Power: 3-cost -- difficult to play without energy support. Only consider with energy-generating relics or cards (Tactician, Adrenaline). Skip if energy generation is not available.
- up my sleeve: Cost reduces by 1 each time played. Against asleep/invincible enemies, play it strictly to reduce its cost for future cycles. Be sure to play the generated Shivs so they exhaust and don't clog.
- accuracy: Power: +4 damage to all Shivs per copy. Base Shiv = 4 dmg -> 8 with 1 copy, 12 with 2 copies. ONLY buffs Shiv cards -- does NOT affect Ricochet, Dagger Spray, or other multi-hit attacks. Stacks: multiple copies multiply value linearly with Shiv generators (Blade Dance, Up My Sleeve, Infinite Blades, Fan of Knives).
- expertise: Draws until your hand has 6 cards. Can be an awkward emergency out; requiring you to spend energy to shrink your hand size first can leave you with 0 energy to play the defensive cards you draw.
- piercing wail: A-tier defense against multi-hit attacks. Exhausts, so save for heavy multi-hit turns. Against Act 3 Boss, can be intentionally discarded to avoid Hunger exhaust and cycle it back.
- blade dance: Premium Shiv engine. Best generator for Accuracy, Fan of Knives, Phantom Blades, Envenom, and Kunai-style scaling. In Shiv decks it is usually stronger than basic attacks or flat-damage filler; upgrade and protect it on remove/transform screens unless you already have redundant generation.
- leg sweep: 2-cost: high Block + applies Weak. Scales with Dexterity for the Block portion. Pounce reduces the next Skill cost to 0 -- play Pounce before Leg Sweep to play it for free.
```

L_3 — Rules

```
## Combat Rules
- Only play cards marked [PLAYABLE]. Cards marked [UNPLAYABLE] will fail.
- Cards show `cost=N` -- you need at least N energy remaining to play them.
- In `plan`, action order is execution order. Earlier plays change your remaining energy, hand composition, and later card costs.
- Include potions in your combat plan when useful (they don't cost energy).
- **Discard effects**: If a card requires discarding (e.g. Survivor), specify with the `"discard"` field. Example: `{"type": "card", "card": "Survivor", "target_index": -1, "discard": "Defend"}`
- **Target priority**: Kill enemies that scale (Strength-stacking, buffing). Against non-scaling enemies, block every attack to minimize HP loss.
```

User prompt: current combat state.

L_2 — Round state

```
## Round 1 State
Energy: 3/3 | HP: 66/70 | Block: 0
Player buffs/debuffs: Dexterity(1): Dexterity improves Block gained from cards.
```

L_2 — Enemies

```
## Enemies
- Soul Nexus [index=0]: HP 234/234, Block 0, Intent: Attack(21) | powers: Weak(1): Weakened creatures deal 25% less damage with Attacks.

Incoming damage: 21 (after block: 21) | Your HP: 66
```

L_2 — Relic counters

```
## Relic Counters
- Letter Opener: 0 -- Every time you play 3 Skills in a single turn, deal 5 damage to ALL enemies.
- Happy Flower: 2 -- Every 3 turns, gain Energy.
```

L_2 — Potions

```
## Usable Potions
Potion slots: 2/3 (1 open)
- [potion_index=0] Powdered Demise [INSTANT] -> targets enemies (target_index required): Enemy loses 9 HP at the end of each of
its turns.
- [potion_index=1] Swift Potion [INSTANT]: Draw 3 cards.
```

L_2 — Piles

```
## Piles
Piles: Draw 20 | Discard 0 | Exhaust 0
Draw (20):
  - Accuracy [1]: Shivs deal 4 additional damage.
  - Accuracy+ [1]: Shivs deal 6 additional damage.
  - Backflip [1]: Gain 5 Block. Draw 2 cards.
  - Blade Dance [1]: Add 3 Shivs into your Hand. Exhaust.
  - Blade of Ink [1]: Add 2 Inky Shivs into your Hand.
  - Byrd Swoop+ [0]: Deal 18 damage.
  - Cloak and Dagger+ [1]: Gain 6 Block. Add 2 Shivs into your Hand.
  - Defend [1]: Gain 5 Block.
  - Deflect [0]: Gain 4 Block.
  - Expertise+ [1]: Draw cards until you have 7 in your Hand.
  - Footwork+ [1]: Gain 3 Dexterity.
  - Knife Trap [2]: Play every Shiv in your Exhaust Pile on the enemy. (Plays 0 Shivs)
  - Leading Strike+ [1]: Deal 6 damage. Add 2 Shivs into your Hand.
  - Leg Sweep [2]: Apply 2 Weak. Gain 11 Block.
  - Phantom Blades [1]: Shivs gain Retain. The first Shiv you play each turn deals 9 additional damage.
  - Piercing Wail [1]: ALL enemies lose 6 Strength this turn. Exhaust.
  - Serpent Form [3]: Whenever you play a card, deal 4 damage to a random enemy.
  - Storm of Steel [1]: Discard your Hand. Add 1 Shiv into your Hand for each card discarded.
  - Up My Sleeve [2]: Add 3 Shivs into your Hand. Reduce this card's cost by 1.
  - Well-Laid Plans [1]: At the end of your turn, Retain up to 1 card.
```

L_3 — Active effects

```
## Key Effects (active this combat)
- Block: Absorbs damage until your next turn, then resets to 0.
- Weak: Target deals 25% less Attack damage for N turns.
- Dexterity: Adds N Block to each Block card.
- Innate: Always in your opening hand.
- Shiv: 0-cost Attack that deals 4 damage and Exhausts. Generated by Blade Dance, Cloak and Dagger, etc.
- Swift N: The first time played, draw N cards.
- Infinite Blades N: At the start of your turn (before draw), add N Shivs into your hand.
```

L_2 — Hand

```
## Hand (7 playable / 7 total)
- Mind Blast (Attack, cost=1) [0 dmg] -> targets enemies: Innate. Deal damage equal to the number of cards in your Draw Pile.
(Deals 28 damage)
    vs Soul Nexus[0]: 28 dmg
- Backflip+ (Skill, cost=1) [8 block] DRAWS: Gain 18 Block. Draw 2 cards.
- Survivor (Skill, cost=1) [8 block]: Gain 18 Block. Discard 1 card.
- Defend (Skill, cost=1) [5 block]: Gain 12 Block.
- Strike (Attack, cost=1) [6 dmg] -> targets enemies: Deal 6 damage.
- Infinite Blades (Power, cost=1): At the start of your turn, add 1 Shiv into your Hand.
  -> generates Shiv (Attack, cost=0, Exhaust): Deal 4 damage. Exhaust.
- Neutralize+ (Attack, cost=0) [4 dmg] -> targets enemies: Deal 4 damage. Apply 2 Weak. Gain 1 energy .
!! DISCARD: Survivor will require discarding. Fill the "discard" field in your plan. Use a list when the card discards multiple
cards.
!! DISCARD RULE: Survivor -- if hand has fewer cards than the discard cost, you only discard what remains (possibly zero).
SEQUENCE: play ALL other cards BEFORE Survivor so it discards 0 cards.

Energy budget: 3E available, fixed-cost total: 6E

CRITICAL RULES:
- Energy RESETS to full each turn. Unspent energy is WASTED.
- Hand cards are DISCARDED at end of turn. Unplayed cards are WASTED.
- Cards DRAWN or CREATED this turn enter your CURRENT hand now. If left unplayed, they are discarded/exhausted/retained by their
own rules at the end of THIS turn -- they do NOT wait for next turn.
- Hand size limit is 10 cards. Current hand is 7/10; draw/add-to-hand effects beyond 10 are lost or fail to enter hand.
- Current Block only matters for the upcoming enemy turn. It does NOT carry into your next turn unless a visible effect explicitly
preserves Block.
- Draw pile order is only a CONDITIONAL forecast. It predicts later draws only if you stop drawing right now; any draw/add-to-hand
effect changes that forecast immediately.
- The `plan` array is executed top-to-bottom. If a card should be played last (for example an X-cost card like Malaise), list it
last.
```

Structured response. The model returned a structured combat plan: use Powdered Demise, play Neutralize+, Mind Blast, Backflip+, Infinite Blades, Survivor, and then end the turn.

E.2 Shop-planning decision

The shop exhibit shows the same interface outside combat. To avoid printing the shared role and generic JSON schema twice, it includes only the shop-specific system additions and the shop user prompt.

Shop-specific system prompt.

L_5 — Deck philosophy

```
## Card & Deck Philosophy
- Evaluate cards along 4 dimensions: **Damage** (kill faster), **Defense** (survive), **Draw** (cycle deck faster), **Energy**
(play more per turn).
- A strong deck needs enough damage to kill bosses in ~10 turns (Act 1 ~= 200 HP, Act 2 ~= 400, Act 3 ~= 600) while surviving.
Damage is the primary constraint -- defense, draw, and energy support damage output.
- **Shops**: Choose whatever gives the biggest power spike for remaining fights -- cards, relics, removal, or potions.
```

L_5 — Two-phase framework

```
## Strategic Deckbuilding: The Two-Phase Framework

You are evaluating card choices. Build your deck around **mechanics and synergies**, not pre-defined archetypes. Deckbuilding has
two distinct phases; knowing your current phase prevents *deck confusion* -- assembling pieces of multiple engines without a
coherent win condition.
```

L_5 — Phase 1

```
### Phase 1 -- Foundation (no engine yet)
Before acquiring a **core scaling engine**, prioritize survival with cards that fit ANY future build:
- **Frontload damage / AoE** -- survive early elites.
- **Generic mitigation** -- efficient block or damage reduction.
- **Cycling / draw** -- hand manipulation and deck thinning.
- **Energy** -- generators that let you play more cards per turn.

*Rule:* Do NOT force a synergy before you hold a card that rewards it. Keep options open.
```

L_5 — Phase 2

```
### Phase 2 -- Commitment (engine acquired)
A **core engine piece** is a card or relic that provides *multiplicative scaling* for a specific keyword, mechanic, or action --
it turns generic cards into a win condition.

Before classifying a card as core, ask: *"Does this card exponentially increase our damage with 2-3 related reward cards and
current deck?"* If yes -> commit.

Once in Phase 2:
1. **Identify the core mechanic** -- what action / keyword / trigger does your engine reward?
2. **Feed the engine** -- prioritize cards that generate, apply, or cycle that mechanic; add enough draw to find the engine fast.
3. **Cover weaknesses** -- add block, AoE, or utility ONLY for what the engine cannot handle itself.
4. **Pivot rule** -- do NOT abandon your engine unless BOTH hold:
   (a) your committed deck has severely insufficient engine pieces (<2 supporting cards), AND
   (b) an offered card is a clearly superior core piece AND solves an immediate survival problem.

Abandoning a partially-built engine wastes every prior pick and leaves two half-engines.
```

L_1 — Note schema

```
## Output: `strategic_note`

Include a `strategic_note` field describing the current deck game plan in one natural-language sentence, under 80 words. Do NOT
write JSON, key-value fields, bullets, or a fresh one-off plan.

The note should be sticky and actionable:
- State whether the deck is still looking for a core engine or already committed.
- Describe how to pilot the deck's strengths: key cards/relics, sequencing, and what the off-turns do.
- Mention the main missing piece and what to avoid adding.
- Only change the engine description when the pivot rule permits.

Good examples:
- "Foundation plan: survive with frontload and efficient block while looking for a real scaling engine; take cheap draw or
high-impact damage, skip narrow synergy pieces."
- "Committed poison plan: retain poison and draw pieces, stack poison on safe burst turns, then defend while passive poison kills.
Needs dex/block scaling; skip off-plan attacks and expensive cards."
```

User prompt.

L_5 — Retrieved skills

```
## Expert Knowledge (retrieved skills)
*Apply these strategies to the current situation. Deviate only with good reason.*


**Silent - Draft and Shop Rules** (seed)
Early rewards must solve damage first, then block, draw, and energy. Premium Silent pickups when offered are Backstab, Pinpoint,
Pounce, Backflip, Footwork, Acrobatics, Escape Plan, Calculated Gamble (usually one copy), Adrenaline, and Tactician. Hidden
Daggers and Corrosive Wave are core shiv/poison cards but are must-picks even outside those archetypes. Expose is a free exhaust
card -- take one copy. Haze, Accuracy, Burst, Shadowmeld, Storm of Steel, Knife Trap, Accelerant, Afterimage, and Tools of the
Trade are strong picks when they fit your current deck building direction. Prepared is strong only after upgrade; Reflex is only
good when discard outlets are already plentiful. If you already have Pounce, expensive skills get better because the next skill
can be free. In shops, try to arrive with 200+ gold when possible; buy order is exceptional colorless cards > Silent cards that
fit the plan > powerful relics that fit the deck building > card removal.

**Deck Building Across the Run** (seed)
Evaluate every card along 4 dimensions: Damage (kill faster), Defense (survive), Draw (cycle deck faster), Energy (play more per
turn). A balanced deck needs all four -- identify which dimension your deck lacks most and prioritize filling that gap. Skip if
the card doesn't clearly improve a weak dimension. Upgrade priority: cards with cost reduction (2->1, 1->0) or doubled effects
first. Do NOT over-thin: keep enough damage AND defense that your deck still functions in every combat. Always think about how
your deck will perform in boss fights -- build enough damage and defense through card synergies. Avoid picking too many
transitional cards that don't combo well. Also avoid having too few damage cards. If you've committed to an archetype (poison,
shiv, etc.), specialize in damage cards for that archetype. Generic damage cards are fine as supplements, but avoid bloating the
deck with cards that don't contribute to your core damage plan.
```

L_4 — Deck insights

```
## Deck Building Insights
*Adapt to your current deck and situation.*

- [Deck Guide: shiv] - **Deck Size & Draw:** Target 23+ cards.
```

L_4 — Card insights

```
## Card-Specific Insights
*Per-card experience -- consider alongside your build plan.*

- skewer: X-cost: deals 7 damage x energy spent. Scales with available energy -- spending 3 energy = 21 damage. Pairs with energy
generation (Adrenaline, Tactician) for higher output.
- precise cut: 0-cost: deals 13 damage minus 2 per other card in hand. Strongest in small hands (1-2 other cards = 9-11 damage for
0 energy). Empty hand = 13 free damage. Pair with hand-emptying effects (Restlessness, Calculated Gamble).
- leg sweep: support for shiv (3 runs): Provided essential Weak application and block to mitigate incoming damage.
- leg sweep: 2-cost: high Block + applies Weak. Scales with Dexterity for the Block portion. Pounce reduces the next Skill cost to
0 -- play Pounce before Leg Sweep to play it for free.
- blade dance: core for shiv (5 runs): Played frequently to generate 3 Shivs for 1 energy, fueling Accuracy and Envenom.
- blade dance: core in 1 win of Repeatedly generating and empowering a large burst of zero-cost multi-hit attacks, with retained
hands and extra card draw allowing the deck to chain several attack sequences in one turn; co-played with Accuracy+, Phantom
Blades, Runic Pyramid
- blade dance: Premium Shiv engine.
```

L_3 — Game knowledge

```
## Game Knowledge
```

L_3 — Card mechanics

```
## Card Mechanics (from game data)
- Skewer: Upgrade: Damage +3
- PreciseCut: Upgrade: Calculationbase +3
- LegSweep: Applies: 2 Weak | Upgrade: Block +3, Weak +1
- BladeDance: Creates: Shiv | Upgrade: Cards +1
- NoxiousFumes: Applies: 2 Noxious Fumes | Upgrade: Poisonperturn +1
- MindBlast: Upgrade: Cost -> 0
- Jackpot: Upgrade: Damage +5
```

L_2 — Shop state

```
## Shop
HP: 64/70 (91%) | Gold: 244
Status: OPEN
Act: 3 | Floor: 38
```

L_2 — Deck

```
## Current Deck (32 cards)
  - Leading Strike+(cost=1) x2: Deal 6 damage. Add 2 Shivs into your Hand.
```

```
  - Strike(cost=1) x2: Deal 6 damage.
  - Byrd Swoop+(cost=0): Deal 18 damage.
  - Neutralize+[Sown](cost=0): Deal 4 damage. Apply 2 Weak. Gain 1 energy .
  - Accuracy(cost=1) x2: Shivs deal 4 additional damage.
  - Footwork+(cost=1): Gain 3 Dexterity.
  - Infinite Blades(cost=1): At the start of your turn, add 1 Shiv into your Hand.
  - Phantom Blades(cost=1): Shivs gain Retain. The first Shiv you play each turn deals 9 additional damage.
  - Serpent Form(cost=3): Whenever you play a card, deal 4 damage to a random enemy.
  - Well-Laid Plans(cost=1): At the end of your turn, Retain up to 1 card.
  - Defend(cost=1) x6: Gain 5 Block.
  - Backflip(cost=1) x3: Gain 5 Block. Draw 2 cards.
  - Backflip+(cost=1): Gain 8 Block. Draw 2 cards.
  - Blade of Ink(cost=1): Add 2 Inky Shivs into your Hand.
  - Cloak and Dagger+(cost=1): Gain 6 Block. Add 2 Shivs into your Hand.
  - Deflect(cost=0): Gain 4 Block.
  - Expertise+(cost=1): Draw cards until you have 7 in your Hand.
  - Knife Trap(cost=2): Play every Shiv in your Exhaust Pile on the enemy.
  - Piercing Wail(cost=1): ALL enemies lose 6 Strength this turn. Exhaust.
  - Storm of Steel(cost=1): Discard your Hand. Add 1 Shiv into your Hand for each card discarded.
  - Survivor(cost=1): Gain 8 Block. Discard 1 card.
  - Up My Sleeve(cost=2): Add 3 Shivs into your Hand. Reduce this card's cost by 1.
```

L_2 — Relics

```
## Relics: Ring of the Snake (At the start of each combat, draw 2 additional cards.), Large Capsule (Upon pickup, obtain 2 random
Relics. Add an additional Strike and Defend to your Deck.), Parrying Shield (If you end a turn with at least 10 Block, deal 6
damage to a random enemy.), Blood Vial (At the start of each combat, heal 2 HP.), Oddly Smooth Stone (Start each combat with 1
Dexterity.), Letter Opener (Every time you play 3 Skills in a single turn, deal 5 damage to ALL enemies.), Vambrace (The first
time you gain Block from a card each combat, double the amount gained.), Byrdpip (Upon pickup, gain the card Byrd Swoop. A Byrdpip
will accompany you in battles.), Biiig Hug (Upon pickup, remove 4 cards from your Deck. Whenever you shuffle your Draw Pile, add a
Soot into your Draw Pile.), Red Mask (At the start of each combat, apply 1 Weak to ALL enemies.), Happy Flower (Every 3 turns,
gain Energy.), Whetstone (Upon pickup, Upgrade 2 random Attacks.), Music Box (Create an Ethereal copy of the first Attack you play
each turn.)
```

L_4 — Relic synergies

```
## Relic Synergies
- **Biiig Hug**: Upon pickup, remove 4 cards from your Deck. Whenever you shuffle your Draw Pile, add a Soot into your Draw Pile.
```

L_4 — Gold budget

```
## Gold Budget Analysis
Affordable: 12 items (7c 2r 3p)
```

L_5 — Shop guide

```
## Guide
Best purchase = biggest power spike for remaining run.

Boss HP: ~600 in ~10 turns -> need ~60 damage/turn.
Estimate your deck's damage output first. If below target, prioritize damage/poison cards.
For scaling attacks, estimate their damage on boss turns 5-10, not just their baseline text.
If a card strengthens an engine you already have, count it as immediate plan support even if future rewards are unknown.
If above target, invest in defense, draw, relics, or save gold.
If card removal costs more than 100g, prefer combat-improving cards, relics, or potions unless removing a Curse.

Review your Build Plan in the Strategic Thread. Prioritize purchases that fill gaps.
```

L_4 — Card notes

```
## Card Notes
- Accuracy: ONLY boosts Shiv cards (Blade Dance outputs, Infinite Blades outputs, Fan of Knives outputs, Cloak and Dagger outputs,
etc.). Does NOT buff other multi-hit attacks like Ricochet or Dagger Spray.
```

L_1 — Task

```
## Your Task
Plan ALL purchases for this shop visit in one decision.
Order matters: items are bought first-to-last. Track gold after each purchase.

For each affordable item you choose NOT to buy, explain why in skipped_items.

Output format:
```
{
  "purchases": [
    {"action": "buy_card|buy_relic|buy_potion|remove_card|discard_potion",
```
```

```
    "item_name": "exact name", "price": <int>,
    "gold_after": <int>, "reason": "..."}
  ],
  "skipped_items": [
    {"item_name": "...", "reason": "..."}
  ],
  "reasoning": "overall strategy",
  "strategic_note": "plain prose current deck game plan, not JSON"
}
```

Rules:
- Gold math must be exact: start with current gold, subtract each price in order.
- If you buy nothing, purchases = [] and list all affordable items in skipped_items.
- Current gold: 244. Double-check gold_after values.
- Schema for card removal purchase entries: action "remove_card", item_name "Card Removal", price 100.

L_3 — Keywords

```
## Keyword Glossary
- Block: Absorbs damage until your next turn, then resets to 0.
- Weak: Target deals 25% less Attack damage for N turns.
- Poison: Loses N HP at the start of its turn, before it acts (attack or buff), then decreases by 1. Bypasses Block.
- Strength: Adds N damage to each Attack hit. Multi-hit cards benefit enormously.
- Dexterity: Adds N Block to each Block card.
- Retain: Stays in hand at end of turn instead of being discarded.
- Exhaust: Removed from combat permanently after use. Thins deck, but you can never play this card again -- bad for damage cards
you want every cycle.
- Innate: Always in your opening hand.
- Shiv: 0-cost Attack that deals 4 damage and Exhausts. Generated by Blade Dance, Cloak and Dagger, etc.
- Inky: When played, apply 1 Weak to target. Powered attacks deal 2 additional damage.
```

L_2 — Items

```
## Items For Sale
- [0] Skewer (XE, 76g) [CAN BUY] Uncommon: Deal 8 damage X times.
- [1] Precise Cut (0E, 78g) [CAN BUY] Uncommon: Deal 13 damage. Deals 2 less damage for each other card in your Hand.
- [2] Leg Sweep (2E, 73g) [CAN BUY] Uncommon: Apply 2 Weak. Gain 11 Block.
- [3] Blade Dance (1E, 24g SALE) [CAN BUY] Common: Add 3 Shivs into your Hand. Exhaust. -> generates: Shiv (Attack, 0E, Exhaust):
Deal 4 damage. Exhaust.
- [4] Noxious Fumes (1E, 78g) [CAN BUY] Uncommon: At the start of your turn, apply 2 Poison to ALL enemies.
- [5] Mind Blast (1E, 84g) [CAN BUY] Uncommon: Innate. Deal damage equal to the number of cards in your Draw Pile.
- [6] Jackpot (3E, 179g) [CAN BUY] Rare: Deal 25 damage. Add 3 random 0 energy cards into your Hand.
- [0] Venerable Tea Set (Relic, Common, 188g) [CAN BUY]: Whenever you enter a Rest Site, start the next combat with an additional
2 Energy.
- [1] Vexing Puzzlebox (Relic, Rare, 309g) [TOO EXPENSIVE]: At the start of each combat, add a random card into your Hand. It's
free to play this turn.
- [2] The Abacus (Relic, Shop, 217g) [CAN BUY]: Whenever you shuffle your Draw Pile, gain 6 Block.
- [0] Attack Potion (Potion, 52g) [CAN BUY]: Choose 1 of 3 random Attack cards to add into your Hand. It's free to play this turn.
- [1] Swift Potion (Potion, 52g) [CAN BUY]: Draw 3 cards.
- [2] Explosive Ampoule (Potion, 51g) [CAN BUY]: Deal 10 damage to ALL enemies.
- [REMOVE] Card Removal (100g) [CAN BUY]
```

L_1 — Shop-plan schema

```
## Decision Format (shop_plan)
Required fields: purchases, skipped_items, reasoning
Optional: strategic_note, note_scope (turn|combat|run), note_triggers (combat|deck_building|routing|all)
strategic_note must be plain prose, not JSON or key-value fields.
```

Structured response. The model returned a shop plan that buys Blade Dance, Leg Sweep, Mind Blast, and Swift Potion, while skipping the remaining affordable items with item-level reasons.