CONDITIONAL MEMORY: MODELING AND TRIGGERING DEFERRED INTENTIONS IN LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Language agents are routinely given memory that retrieves the past, yet reliable assistance also requires prospective memory: holding a deferred intention until a future cue and executing it amid ongoing work, including after cancellations and reschedules. This skill remains unsolved. Grafting retrospective stores such as Mem0 or A-Mem onto a frozen model falls below a no-store baseline, because similarity retrieval surfaces obsolete notes rather than a revisable due set. We argue that the bottleneck is the object of memory, not a stronger retriever. We present ProMem, an inference-time scaffold whose core is a Prospective Intention Store: typed records that couple a trigger, an action, watched channels, and a lifecycle state. Form and an Update Judge write and revise those records; due filtering and completion marking run in code, with no backbone fine-tuning. The agent therefore sees a compact board of time-due items, event-due items, and unchecked channels, rather than an ever-growing transcript or an unfiltered note dump. On the released PM-Bench week, ProMem reaches $83.6\pm5.7\%$ Set-F1 when averaged over seven frozen backbones (three runs each), above the no-store baseline, Mem0, A-Mem, and the prompting scaffolds we re-run. It leads mean Set-F1 on five of seven models, while a strong Baseline remains competitive on Gemini-3.8-flash and Kimi-k3. Ablations show that replacing due rules with similarity retrieval, or applying those rules to raw text, removes most of the gain; removing lifecycle status mainly increases false alarms. Our contributions are a formalization of prospective memory as operations over conditional records, a training-free instantiation of that abstraction, and evidence that structure, rather than retrieval alone, is what makes deferred execution reliable.

1 INTRODUCTION

Language agents that plan, call tools, and act over many turns depend on external memory (Yao et al., 2023; Schick et al., 2023; Wang et al., 2024). The dominant design is *retrospective*: write notes during interaction and later retrieve those relevant to a query (Lewis et al., 2020; Packer et al., 2023; Chhikara et al., 2025; Xu et al., 2025). Success is measured by whether the right past fact surfaces under $\mathrm{sim}(e, q)$, answering what happened from history (Maharana et al., 2024; Wu et al., 2024).

Reliable assistance also requires a different competence. Users defer instructions—take medicine at 21:00, book when a portal opens, reply once mail arrives, cancel if plans change—and success is acting at the right future moment, or refraining when an intention has been canceled or a menu item is only a lure. Cognitive science calls this prospective memory: forming a delayed intention, monitoring for a time- or event-based cue during ongoing work, and then executing or withdrawing the action (Einstein & McDaniel, 1990; McDaniel & Einstein, 2007). PM-Bench adapts the Virtual Week paradigm to tool-using agents and scores the due set at each step (Rendell & Craik, 2000; Liu & Gabriel, 2026). Strong models remain unreliable on it. The best published scaffold reaches 65.1% Set-F1; under the seven frozen backbones in Section 4, the macro-averaged todo-ledger and heartbeat scores are 41.6 ± 22.7 and 71.3 ± 10.9. TriggerBench shows the same split, with near-ceiling retrospective accuracy where proactive intervention collapses (Zhang et al., 2026).

Current memories do not close the gap. Prompting scaffolds such as todo ledgers and heartbeats trade missed dues against false alarms (Liu & Gabriel, 2026). Retrospective stores such as Mem0

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

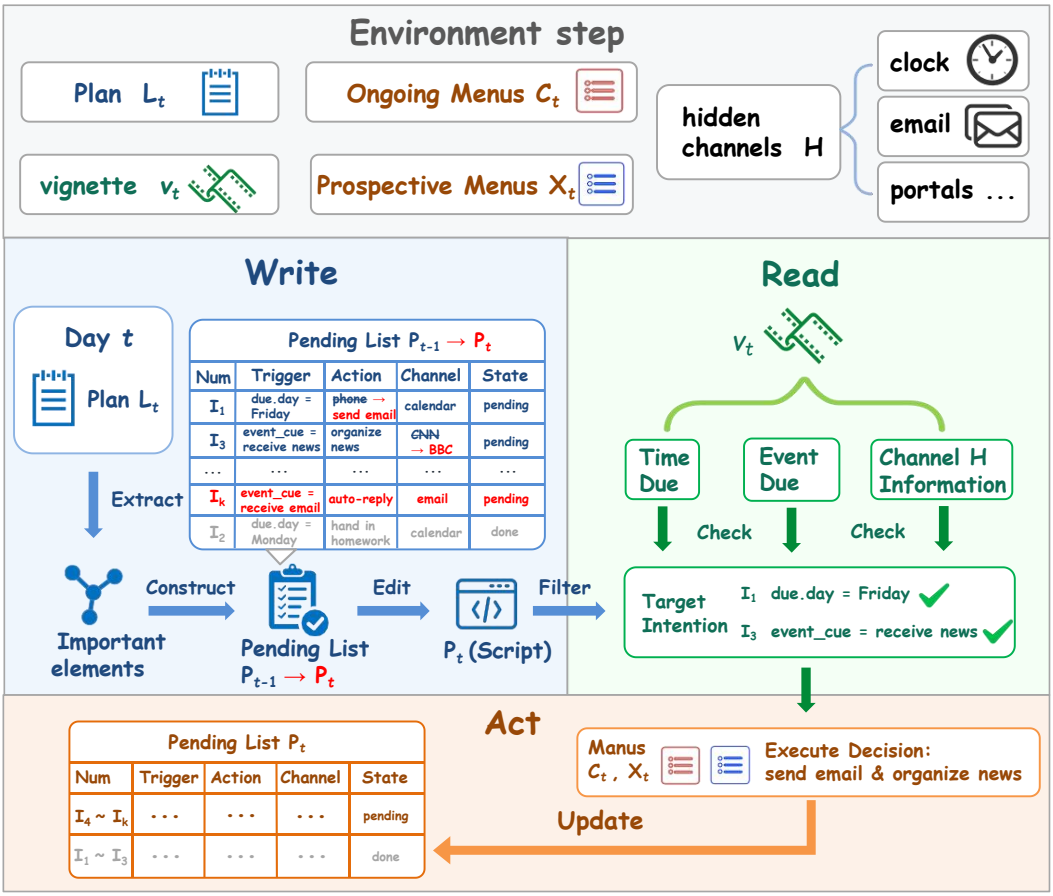

Figure 1: One step of ProMem. Form and an Update Judge write the Prospective Intention Store. Code then keeps today's pending rows and builds a board of TIME-DUE, EVENT-DUE, and CHECK items for a compact context. The backbone may query a hidden channel, which rebuilds the board (dashed), or choose; code marks matched rows done. If a timed row is pending and the clock is unknown, the controller forces `check_time`.

and A-Mem, placed on the same frozen backbone, fall below a no-store baseline, because similarity retrieval surfaces obsolete notes rather than a revisable due set (Chhikara et al., 2025; Xu et al., 2025). Writing an intention down as ordinary text, and expecting a retriever or a longer context to notice when it comes due, fails on the cases PM-Bench isolates: hidden channels, lures, and cancellations that must persist across days (Lewis et al., 2020; Liu et al., 2024).

We present **ProMem**, an inference-time scaffold whose core is a Prospective Intention Store (PIS; Figure 1). Each deferred intention is a typed record with a trigger, an action, watched channels, and a lifecycle state. Form and an Update Judge write and revise those records from the day plan and the current scene; the judge does not decide what is due. Deterministic code keeps today's pending rows and builds a board of time-due items, event-due items, and channels still to check, then injects that board into a compact step context. The backbone chooses an ongoing activity and a prospective set, or queries a hidden channel and rebuilds the board. If a timed row is pending and the clock is unknown, the controller forces a clock query. After a choice, code marks the matched rows done. The backbone is not fine-tuned.

On the released PM-Bench week, ProMem reaches 83.6 ± 5.7 Set-F1 when averaged over seven frozen backbones, above a no-store baseline at 74.5 ± 10.6 and heartbeat at 71.3 ± 10.9. It leads mean Set-F1 on five of seven models; Baseline remains strong on Gemini-3.8-flash and Kimi-k3. Replacing the due rules with similarity retrieval drops Set-F1 to 58.3.

In summary, our contributions are as follows:

1. We formalize prospective memory for tool-using agents as operations over typed conditional records, with due-set Set-F1 as the objective rather than retrospective similarity retrieval.
2. We instantiate this abstraction as ProMem, a training-free scaffold that writes a Prospective Intention Store and filters a due board in code.
3. We evaluate it on PM-Bench across seven frozen backbones, against retrospective memories and re-run prompting scaffolds, with ablations, cost accounting, and a 30-day stress test.

We will release code, prompts, and store schemas upon acceptance.

2 Related Work

LLM agents and long-horizon interaction. ReAct-style reason–act loops, tool-augmented LMs, and cognitive architectures for language agents establish the setting in which memory becomes necessary (Yao et al., 2023; Schick et al., 2023; Patil et al., 2023; Qin et al., 2023; Sumers et al., 2024; Park et al., 2023; Shinn et al., 2023; Wang et al., 2023; Liu et al., 2023; Huang et al., 2024; Wang et al., 2024). Long-context analyses show that simply enlarging the window does not guarantee reliable use of buried constraints (Liu et al., 2024). We study an orthogonal failure mode: not "find the fact," but "fire the deferred action when the cue is true."

Retrospective agent memory. Production and research stacks store episodic or semantic notes and retrieve them under a query—MemGPT/Letta, MemoryBank, Mem0, A-Mem, MemoryOS, MemOS, LightMem, Zep, HippoRAG, and related read–write memories (Packer et al., 2023; Zhong et al., 2024; Chhikara et al., 2025; Xu et al., 2025; Kang et al., 2025; Li et al., 2025; Fang et al., 2025; Rasmussen et al., 2025; Gutiérrez et al., 2024; Hu et al., 2023; Modarressi et al., 2023; Asai et al., 2024; Lewis et al., 2020; Zhang et al., 2024). Benchmarks such as LoCoMo, LongMemEval, and newer agent-native suites probe retention, updates, and temporal QA over long histories (Maharana et al., 2024; Wu et al., 2024; Zhou et al., 2026; Zhao, 2026). Reflective prospect/retrospect managers improve personalized dialogue (Tan et al., 2025) but still target answer quality rather than due-set Set-F1 under hidden channels. These lines optimize $\mathrm{sim}(e, q)$; they do not maintain revisable $(\varphi, \alpha, W, \sigma)$ intentions or score whether a trigger holds in V_t.

Prospective memory. Cognitive work distinguishes event- and time-based delayed intentions and measures monitoring cost under ongoing tasks (Einstein & McDaniel, 1990; Einstein et al., 2005; Smith, 2003; McDaniel & Einstein, 2007; Rendell & Craik, 2000). PM-Bench adapts Virtual Week for tool-using LLM agents with Set-F1 over due sets, lures, cross-day carry-over, and update-sensitive slices; it reports 65.1% Set-F1 for its best scaffold (Liu & Gabriel, 2026). TriggerBench and related studies corroborate that proactive PM lags matched retrospective probes (Zhang et al., 2026; Mittal, 2026). PM-Bench evaluates prompting, todo ledgers, heartbeats, and hierarchical union-query, but does not provide a typed intention store whose due tests are enforced in code. ProMem is complementary: it keeps frozen backbones and places structure in the Prospective Intention Store—the object of memory—rather than in a larger episodic bank or a fixed polling schedule.

3 Methodology

ProMem treats deferred execution as operations over typed conditional records rather than as another retrieval problem. We first formalize the PM-Bench environment and the intention representation in Section 3.1, where each deferred intention is stored as a typed record outside the context window. We then describe how intentions enter and leave the store in Section 3.2, and how deterministic filters assemble a due board in Section 3.3. Finally, we describe action selection and summarize the full one-step procedure in Section 3.4.

3.1 Problem Formulation

Environment protocol. We follow the PM-Bench Virtual Week protocol (Liu & Gabriel, 2026). Time advances in discrete steps $t = 1, \ldots, T$ (released $T = 80$). At step t the observation has three parts: a vignette v_t, an ongoing-activity menu $\mathcal{C}_t = \{A, B, C\}$, and an anonymous prospective menu X_t. Items in $\mathcal{C}_t$ are parallel activities in the current scene; the agent must pick exactly one

Table 1: PIS record. Slots of φ and W may be empty.

Field	Content	Values
`action`	α	menu-aligned verb phrase
`due.day` / `due.time`	calendar / clock of φ	weekday / `HH:MM`
`event_cue`	narrative part of φ	text
`watch`	W	channel names
`status`	σ	`pending` / `done` / `canceled`

$c_t \in \mathcal{C}_t$ to advance the simulation, and that choice is unscored. Each item in X_t is a handle `task_N` paired with a verb phrase, mixing true tasks with lures. Handles are renumbered each day and are not stable identifiers across days.

State channels $\mathcal{H}$ (clock, email, portals, ...) are hidden from v_t; the model must select a concrete channel to query. A single query is $q \subseteq \mathcal{H}$; write $q_t \subseteq \mathcal{H}$ for the channels already queried at this step. Evidence starts as $V_t \leftarrow v_t$; each query updates $q_t \leftarrow q_t \cup q$ and $V_t \leftarrow V_t \oplus q$. The agent must then submit

$$a_t = (c_t, \hat{D}_t), \qquad \hat{D}_t \subseteq X_t.$$

Only $\hat{D}_t$ is scored. The official due set is

$$D_t = \{x \in X_t : x \text{ is pending and its condition holds at step } t\}.$$

An event task becomes due when the corresponding cue appears in v_t or in an obtained channel reply; a time task becomes due when the simulated clock equals the target time (or falls in an allowed window). Cancellations and old conditions superseded by a reschedule or override are invalid. $\hat{D}_t$ is compared with D_t by trajectory Set-F1 (Section 4).

Intention representation. To keep deferred intentions from being mixed with retrospective memory, the agent maintains them in a separate external store $\mathcal{P}_t$. A deferred intention involves four kinds of information: when to act (time or event), what to execute (action), what to monitor (watched channels), and whether the record is still valid (lifecycle). Storing this as prose entangles the fields: a program cannot test whether a time has arrived or a channel has been read, and the entire string must be handed to the language model. We therefore record each intention as $I = (\varphi, \alpha, W, \sigma)$: trigger φ, intended action α, watch $W \subseteq \mathcal{H}$, and lifecycle $\sigma \in \{\text{pending}, \text{done}, \text{canceled}\}$. Natural-language intentions are written into the slots of Table 1 (φ and W may be empty).

Intention filtering, action choice, and store update are given by

$$\iota_t = g(\mathcal{P}_t, V_t), \qquad \hat{D}_t = \pi(\iota_t, V_t), \qquad \mathcal{P}_{t+1} = F(\mathcal{P}_t, V_t, \hat{D}_t). \tag{1}$$

Filtering g is deterministic matching of the current state against due conditions; action choice π is left to the backbone; store update F is carried out by the model on the PIS.

3.2 Write Path

Writing to the PIS has two stages: forming an intention (**Form**) and revising it (**Update**).

Form. On first appearance, standing tasks and the daily plan L_t are rewritten by the model into parseable lines (`<action> at HH:MM`, `when <cue>`, `On <day>, ...`). Code then splits each line into action, day, time, event, and watch slots in $\mathcal{P}_t$; the watch set is the channel keywords mentioned in the sentence.

Update. An Update Judge, implemented as a model call, reads the current vignette and any channel replies already returned in the step. It cancels ($\sigma \leftarrow$ canceled), reschedules (edits `due`), or overrides (rewrites `event_cue`) pending rows, and may insert new intentions. The same call emits `scene_cues`: vignette facts rewritten in `event_cue` style, later used for similarity matching against event triggers.

3.3 Read Path

After an intention is read from the PIS, deterministic procedures filter it along several dimensions and inject still-valid, condition-matched records onto the board; the model then makes the final choice. The pipeline is eligibility, time matching, channel check, event matching, then injection.

Eligibility. Completed or canceled records would interfere with later tests, so the first filter keeps only pending intentions. A coarse calendar check then retains records that are not time-triggered (`due.day` empty) or whose trigger day is today:

$$\mathcal{P}_t^{\text{today}} = \big\{ I \in \mathcal{P}_t : \sigma(I) = \text{pending},\ \texttt{due.day}(I) \in \{\emptyset, \text{today}\} \big\}. \tag{2}$$

Later clock, channel, and event tests all run on this set; other rows stay in $\mathcal{P}_t$ but do not enter the board at this step.

Time matching. Because the vignette may contain time-like distractors, only a channel reply of the form `Time: HH:MM` is treated as a trusted clock. When that clock is present and `due.time` of a record in $\mathcal{P}_t^{\text{today}}$ equals it, the record is marked `TIME-DUE`:

$$\text{TIME-DUE}_t = \{ I \in \mathcal{P}_t^{\text{today}} : \texttt{due.time}(I) \neq \emptyset,\ \text{clock}(V_t) = \texttt{due.time}(I) \}. \tag{3}$$

If no trusted clock has been obtained but $\mathcal{P}_t^{\text{today}}$ still contains a timed record, the controller issues a mandatory clock query and repeats time matching (Algorithm 1, step 6).

Channel check. The model often treats the vignette as sufficient and rarely queries hidden channels. We therefore attach a watch set when an intention is first stored, and take the union of unread watches on the current candidates:

$$\text{CHECK}_t = \Big(\bigcup_{I \in \mathcal{P}_t^{\text{today}}} W(I) \Big) \setminus q_t. \tag{4}$$

This set is injected onto the board as the object of a query.

Event matching. A programmatic score compares `event_cue(I)` with matching text $G_t(I)$ (scoring details in Appendix D), built from the Update Judge's `scene_cues`, the raw vignette v_t, and replies in V_t from channels $W(I)$:

$$\mathcal{E}_t^0 = \{ I \in \mathcal{P}_t^{\text{today}} : \text{sim}(\texttt{event_cue}(I), G_t(I)) \geq \tau \}, \tag{5}$$

$$\text{EVENT-DUE}_t = \{ I \in \mathcal{P}_t^{\text{today}} : \exists\, J \in \mathcal{E}_t^0,\ \texttt{event_cue}(I) \sim \texttt{event_cue}(J) \}. \tag{6}$$

Hits that share a cue are kept as a group so one recognized trigger does not hide the rest.

Injection. The two due sets and the unread-channel set are written onto the model's board:

$$\iota_t = \big(\text{TIME-DUE}_t,\ \text{EVENT-DUE}_t,\ \text{CHECK}_t \big). \tag{7}$$

This is the candidate list shown to π, from which the model chooses further.

3.4 Action Selection and Summary

The decision model sees the system instructions, standing-task titles, the day heading, the injected board (ι_t, V_t), and the prospective menu X_t. It may query a channel; the reply is merged into V_t and the board is rebuilt before the final choice. It then submits $(c_t, \hat{D}_t)$, and code marks the selected records `done` in the PIS. The full algorithm is given in Appendix A.

We highlight the key features of ProMem as a test-time scaffold:

- **Structured intention store.** Deferred intentions remain in $\mathcal{P}_t$ as $I = (\varphi, \alpha, W, \sigma)$, with action, trigger, watch channels, and lifecycle in separate slots. This yields a clean filtering process and keeps the same action under different conditions as distinct records.
- **Rule-based filtering.** Each step evaluates today's pending records by comparing the clock, watched channels, and event cue separately, and injects due records onto the board as TIME-DUE_t, EVENT-DUE_t, and CHECK_t.
- **Lifecycle maintenance.** Form and Update write or revise still-valid intentions; after choose, code marks matching records `done`. Cancel, reschedule, and override edit $\mathcal{P}_t$, so later steps read the revised φ and σ, not the stale morning plan.

4 Experiments

We evaluate deferred-intention execution with frozen language-model parameters. The experiments address four questions: overall accuracy (Section 4.2), the contribution of each mechanism (Section 4.3), robustness across frozen backbones (Section 4.4 and Figure 2), and the size of the monitoring cost (Sections 4.5–4.6). Averaged over seven frozen backbones, ProMem reaches 83.6 ± 5.7 Set-F1, above Baseline and the prompting scaffolds; the gain is concentrated in time-triggered execution and in maintaining state across days. Ablations show that typed triggers, due computation, and lifecycle maintenance are all necessary. The cost is a higher query rate and a higher false-alarm rate, and long-horizon retention remains a bottleneck.

4.1 Experiment Setup

Benchmark and protocol. We use the released PM-Bench Virtual Week (Liu & Gabriel, 2026): 7 simulated days, 80 interaction steps, and 81 scored tasks (57 event-triggered and 24 time-triggered), including cross-day intentions, cancellations, reschedules, and overrides. At each step, the agent observes the current vignette and task menu, may query hidden state or time, and submits the set of tasks to execute. All methods share the scenario, task menu, ground-truth due sets, evaluator, and frozen backbone; only the inference-time scaffold differs. We evaluate seven frozen backbones: DeepSeek-v4-flash, Qwen3.8-flash, Gemini-3.8-flash, GLM-5.3-flash, Claude-opus-5, GPT-5.6-luna, and Kimi-k3. Each backbone$\times$method pair is three complete-week trajectories. Table 2 reports the mean$\pm$std of the seven per-backbone means; per-backbone tables are in Appendix E.

Metrics. The primary metric is the official PM-Bench F1 score (Liu & Gabriel, 2026), computed by comparing the model's choice $\hat{D}_t$ with the official due set D_t at each step. We also report the finer PM-Bench slices: event-triggered hit (Event hit), time-triggered hit (Time hit), miss rate after revision (Update miss), miss rate on cross-day tasks (Cross-day miss), and false alarms per step (FA/step).

Comparators. Baseline decides from the full model context. **Mem0** (Chhikara et al., 2025) extracts factual entries from context into an external memory and, at each step, retrieves related facts by similarity. **A-Mem** (Xu et al., 2025) organizes context into attributed, linkable memory notes (`content`, `keyword`, `tags`, `context`, `links`, ...). A new write is linked to existing notes; a query recalls the matched notes and their neighbors, which are then judged by the language model. **Heartbeat** is the strongest published PM-Bench scaffold and adds periodic reminders; **todo-ledger** is the next-best published scaffold and supplies an in-context list of recorded tasks (Liu & Gabriel, 2026). **ProMem** stores structured intentions in the Prospective Intention Store (PIS; Section 3), filters them by time, event cue, and lifecycle status to build a due board, and then lets the model decide whether to query or execute.

4.2 Main Results on the Released Week

Method	Set-F1↑	Event hit↑	Time hit↑	Update miss↓	Cross-day miss↓	FA/step↓
Baseline	74.5±10.6	65.2±12.8	59.7±23.4	40.2±24.1	36.7±30.7	5.9±2.9
Mem0	65.6±14.4	59.3±12.3	52.2±23.6	62.4±26.8	60.5±21.0	14.3±5.1
A-Mem	66.7±11.1	57.8±6.3	56.2±21.8	64.5±24.4	40.2±18.6	10.7±3.2
Heartbeat	71.3±10.9	68.3±15.3	55.6±21.6	45.5±26.8	36.7±30.7	5.7±2.7
Todo-ledger	41.6±22.7	36.3±21.3	30.2±26.2	69.8±32.6	59.2±28.4	**2.7±1.5**
PMA	64.6±11.4	58.2±10.4	52.4±24.8	66.7±20.6	58.5±22.6	11.8±4.7
ProMem	**83.6±5.7**	**73.1±7.1**	**83.7±6.5**	**20.6±7.7**	**24.5±13.6**	10.3±2.0

Table 2: Released-week results averaged over seven frozen backbones. Each backbone$\times$method cell is first averaged over three runs; the table then reports mean$\pm$std of those seven backbone means. Per-backbone details are in Appendix E.

Performance of ProMem. As Table 2 and Figure 2 show, ProMem reaches 83.6 ± 5.7 Set-F1 across backbones, 9.1 points above Baseline and above Heartbeat and Todo-ledger on the same macro average. On individual backbones (Figure 2), ProMem is first on five of seven models; on Gemini-3.8-flash and Kimi-k3 a strong no-store Baseline remains competitive or ahead.

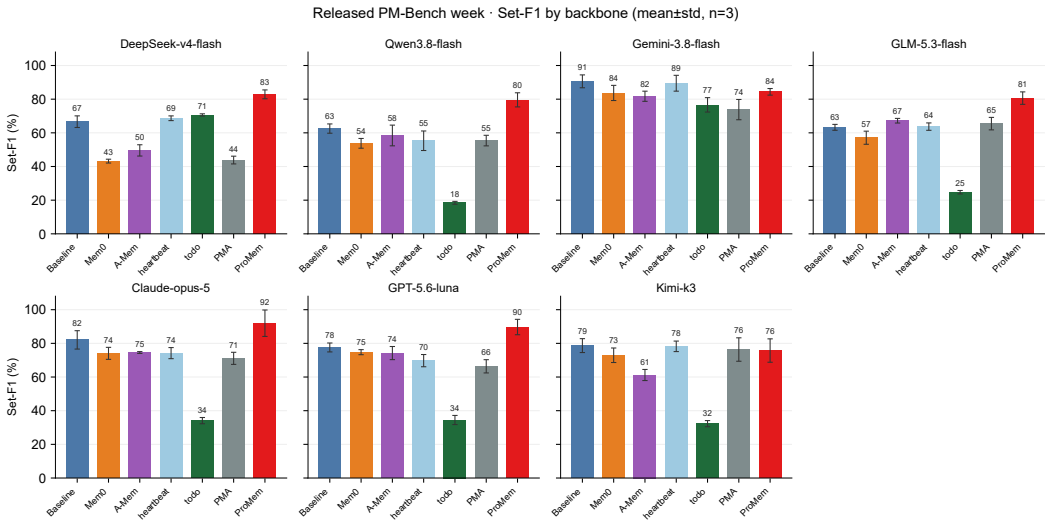

Figure 2: Set-F1 (%) on the released PM-Bench week for seven frozen backbones (mean±std over three complete-week runs per backbone×method).

Source of the gain. The gain is concentrated in time-triggered execution (time hit 83.7 ± 6.5 versus 59.7 ± 23.4 for Baseline) and in maintaining state across days and after revision (Cross-day miss 24.5 ± 13.6, Update miss 20.6 ± 7.7, both well below Baseline). These are the two core skills of prospective memory.

Retrospective memory underperforms. Mem0 and A-Mem do worse than the no-store baseline on this task: Set-F1 falls by 8.9 and 7.8 points relative to Baseline on the macro average. Their retrieval is limited to semantic similarity, so it can surface unrelated distractors while failing to retrieve information that is not sufficiently similar to the current query. When the recalled notes are not structured, the model cannot reliably judge when an intention is due.

4.3 Ablation Analysis

To isolate the role of each ProMem component, we run three ablations: replace rule-based filtering at read-out with semantic-similarity retrieval (**PIS-RAG**); remove lifecycle status (**w/o status**); and remove structured trigger fields, applying the same rules to raw text (**Raw-text due**).

Table 3: ProMem ablations on the released week with DeepSeek-v4-flash.

Variant	Set-F1↑	Hit↑	Event↑	Time↑	Update miss↓	Cross-day miss↓	FA/step↓
ProMem	**85.7**	**81.5**	**75.4**	**95.8**	11.1	**0.0**	7.5
PIS-RAG	58.3	45.7	63.2	4.2	77.8	28.6	10.0
w/o status	80.0	**81.5**	**75.4**	**95.8**	**0.0**	14.3	18.8
Raw-text due	67.7	54.3	42.1	83.3	77.8	100.0	**5.0**

PIS-RAG drops Set-F1 by 27.4 points and time hit to 4.2, so semantic relevance is not a substitute for explicit due computation. **w/o status** preserves hit rates but raises FA/step from 7.5 to 18.8: without lifecycle status, the model cannot tell whether a task is already done or canceled, resulting in repeated triggering and monitoring. **Raw-text due** drops Set-F1 to 67.7, event hit to 42.1, and Cross-day miss to 100.0, showing that the filtering rules depend on a structured intention store.

Taken together, these results show that rule-based due filtering, lifecycle maintenance, and structured intention storage are effective for prospective-intention handling and depend on one another: the best result appears when the three mechanisms are used jointly.

4.4 Robustness across Backbones

Robustness across backbones. Figure 2 compares all seven backbones under the same protocol. ProMem attains the highest mean Set-F1 on five of seven models, with large gains on DeepSeek-v4-flash, Qwen3.8-flash, GLM-5.3-flash, Claude-opus-5, and GPT-5.6-luna. On Gemini-3.8-flash and

Kimi-k3, Baseline remains strong; ProMem is still competitive and keeps a high time hit. We treat long-horizon robustness separately in the 30-day stress test (Section 4.6).

4.5 Compute and Monitoring Cost

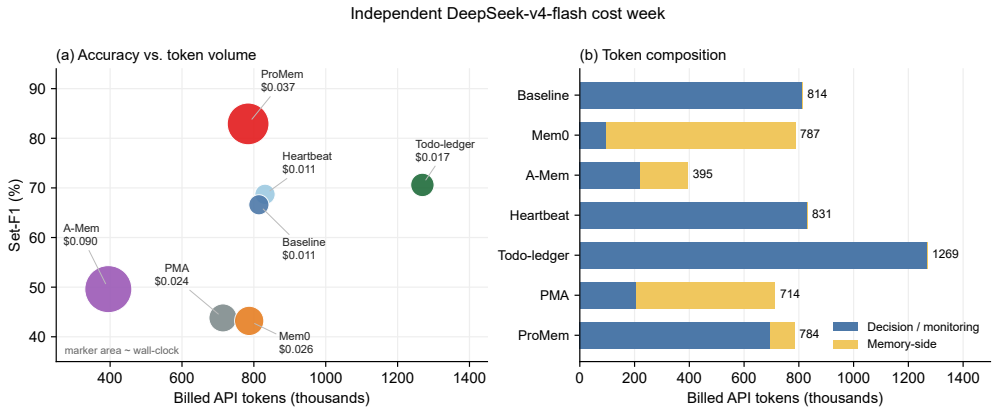

Figure 3: Efficiency on an independent DeepSeek-v4-flash cost week. (a) Set-F1 (released-week three-run means) versus billed tokens; marker area scales with wall-clock. (b) Token composition: Decision / monitoring versus Memory-side.

Table 4: Billed-token composition on the DeepSeek-v4-flash cost week (thousands of tokens).

Method	Total	Decision / monitoring	Memory-side
Baseline	814	814	0
Mem0	787	95	692
A-Mem	395	220	175
Heartbeat	831	831	0
Todo-ledger	1269	1269	0
PMA	714	207	507
ProMem	784	694	90

Token budget. On the cost week, ProMem uses 784k billed tokens, of which 694k are Decision / monitoring and 90k are Memory-side (Form rewrite and Update Judge; Table 4, Figure 3b). Its total is close to Baseline (814k), Heartbeat (831k), and Mem0 (787k), while Todo-ledger is higher (1269k) because the ledger enlarges every decide prompt, and A-Mem is lower (395k). PMA resembles Mem0 in mix (207k decision / 507k memory) at a slightly smaller total (714k). ProMem's peak context remains much smaller than Baseline's (2,484 versus 17,260).

ProMem therefore does not buy accuracy with a larger token budget: the token total is on par with Baseline, and because the injected intentions have already been filtered, the per-call context is an order of magnitude shorter. The accompanying cost is more calls and a higher bill (401 calls / $0.037, versus 88 / $0.011 for Baseline), which follows from the due-check pipeline and from more frequent state queries.

Wall-clock time. On the same cost week, Baseline, Mem0, ProMem, and A-Mem take 89s, 241s, 516s, and 655s (1.5, 4.0, 8.6, and 10.9 minutes). ProMem is about 5.8× slower than Baseline but still faster than A-Mem. Per-call latency is comparable to Baseline (p50 0.98s versus 0.96s); the extra time comes from more calls (401 versus 88), not from slower individual requests.

The longer wall-clock is therefore the result of proactive monitoring splitting one environment step into many short calls. The time cost is higher than that of a context-only Baseline, but still lower than that of A-Mem, whose individual requests are slower (p95 3.83s).

4.6 Month-Long Stress Test

Protocol. Cross-day tasks are one of the harder cases for prospective intention. To examine this slice more fully, we modify the PM-Bench setting and construct a separate 30-day extension with 333 interaction steps and 334 tasks, including dedicated cross-day items with delays of at least seven days, paraphrased event cues, lures, and 48 lifecycle updates (Appendix F). The released PM-Bench

phrasing is comparatively simple and semantically explicit. To better test how the method would fare under more realistic wording, we also diversify the phrasing of prospective intentions on this extension, which makes writing into the store more demanding.

Table 5: Results on the 30-day PM-Bench extension with DeepSeek-v4.

Mechanism	Set-F1↑	Event Hit↑	Time Hit↑	Cross-day miss↓	Update miss↓	FA/step↓
Baseline	63.2	51.2	44.5	55.6	69.2	**2.7**
Mem0 (Chhikara et al., 2025)	54.8	53.7	9.5	61.1	82.1	17.4
A-Mem (Xu et al., 2025)	56.6	54.0	18.7	72.2	94.9	28.5
Heartbeat	64.3	57.5	35.2	66.7	58.4	17.4
Todo-ledger	52.6	57.1	32.8	**27.8**	85.4	4.3
PMA	51.8	55.3	10.6	77.8	88.2	8.2
ProMem	**69.2**	**68.9**	**74.8**	88.9	**10.3**	28.8

Table 6: Results on the 30-day PM-Bench extension with Claude.

Mechanism	Set-F1↑	Event Hit↑	Time Hit↑	Cross-day miss↓	Update miss↓	FA/step↓
Baseline	64.1	58.3	43.2	**61.1**	68.4	**3.5**
Mem0 (Chhikara et al., 2025)	58.2	54.6	15.4	**61.1**	84.8	14.7
A-Mem (Xu et al., 2025)	57.8	55.1	19.6	77.8	93.7	27.6
Heartbeat	60.4	53.6	39.1	**61.1**	55.2	17.8
Todo-ledger	25.7	40.2	19.5	83.3	81.3	5.1
PMA	55.3	53.4	11.3	66.7	89.1	7.9
ProMem	**70.4**	**71.6**	**75.4**	77.8	**7.8**	26.1

Comparative result and residual bottleneck. Despite the harder horizon, ProMem remains ahead of the other six mechanisms on the metrics that track correct deferred execution. On DeepSeek-v4 it leads Set-F1 (69.2 vs. 64.3 for the next-best Heartbeat and 63.2 for Baseline), Event Hit (68.9), Time Hit (74.8), and Update miss (10.3, an order of magnitude below Mem0 / A-Mem / PMA). On Claude the same pattern holds: Set-F1 70.4 (vs. 64.1 Baseline), Event Hit 71.6, Time Hit 75.4, and Update miss 7.8. Retrospective memories and prompting scaffolds do not overtake ProMem once the calendar grows; most of them stay in the mid-50s Set-F1 with Update miss above 80. At the same time, ProMem's Cross-day miss (88.9 / 77.8) and FA/step (28.8 / 26.1) stay high, so absolute long-horizon retention and monitoring precision remain open (Section 6). Scores on this extension are not comparable to the released week.

5 Conclusion

ProMem treats deferred execution as operations over typed records $(\phi, \alpha, W, \sigma)$ in a Prospective Intention Store. Form and an Update Judge write that store; code filters a due board and marks completed rows. On the released PM-Bench week, ProMem reaches 83.6 ± 5.7 Set-F1 averaged over seven frozen backbones, above the no-store baseline and all re-run prompting scaffolds, and leads on five of seven models. The 30-day stress test preserves this ranking on Set-F1, Event Hit, Time Hit, and Update miss, while exposing open problems in cross-day retention and false-alarm control.

6 Limitations and Future Work

Long-horizon retention and monitoring precision remain unsolved. On the 30-day extension, ProMem still records high Cross-day miss (88.9 on DeepSeek-v4, 77.8 on Claude), up sharply from the released PM-Bench week (0.0 on DeepSeek-v4, 28.6 on Claude), where cross-day cues are fewer and more explicit. Despite achieving the highest Set-F1 among all evaluated mechanisms, closing this retention gap remains a primary target for future work.

Dataset limitations. At the time of writing, PM-Bench is the only open-source prospective memory benchmark (the widely used TriggerBench was not released when this paper was prepared). Although we extend PM-Bench with a 30-day scenario, the diversity of evaluation settings remains limited. We hope this work draws attention to prospective memory as a critical problem for LLM agents and encourages the community to develop additional benchmarks and studies.

AI USE STATEMENT

In this work, we used generative AI tools for drafting and polishing English text, suggesting LaTeX formatting, assisting with literature organization. All methods and experiments in the paper were thought and implemented by humans.

ETHICS STATEMENT

This work studies agent memory scaffolds on simulated benchmarks (PM-Bench Virtual Week and synthetic conditional-instruction dialogues). It does not involve human-subject experiments, personally identifiable data collection, or deployment of autonomous systems in high-stakes settings. Potential dual-use concerns are limited; the proposed components (intention stores, monitors, due-set scorers) are evaluation-oriented scaffolds.

REPRODUCIBILITY STATEMENT

We will release the code of the paper methodology and the replication method when publishing the paper. The data constructed in the Section 4.6 will also be published synchronously.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *International Conference on Learning Representations*, 2024.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Gilles O. Einstein and Mark A. McDaniel. Normal aging and prospective memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 16(4):717–726, 1990.

Gilles O. Einstein, Mark A. McDaniel, Ruthann Thomas, Sara Mayfield, Hilary Shank, Nova Morrisette, and Jennifer Breneiser. Multiple processes in prospective memory retrieval: Factors determining monitoring versus spontaneous retrieval. *Journal of Experimental Psychology: General*, 134(3):327–342, 2005.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. LightMem: Lightweight and efficient memory-augmented generation. *arXiv preprint arXiv:2510.18866*, 2025.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. *Advances in Neural Information Processing Systems*, 2024.

Chenxu Hu, Jie Fu, Chenzhuang Du, Siqi Luo, Junbo Zhao, and Hang Li. Chatdb: Augmenting LLMs with databases as their symbolic memory. *arXiv preprint arXiv:2306.03901*, 2023.

Xu Huang, Weiwen Liu, Xiaolong Chen, et al. Understanding the planning of LLM agents: A survey. *arXiv preprint arXiv:2402.02716*, 2024.

Jiazheng Kang, Mingming Ji, Taize Zhao, and Fei Li. Memory OS of AI agent. *arXiv preprint arXiv:2506.06326*, 2025.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Zhiyu Li, Shenyu Song, et al. MemOS: A memory OS for AI system. *arXiv preprint arXiv:2507.03724*, 2025.

Genglin Liu and Saadia Gabriel. PM-Bench: Evaluating prospective memory in LLM agents. *arXiv preprint arXiv:2607.12385*, 2026.

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

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaijie Men, Kejuan Yang, et al. AgentBench: Evaluating LLMs as agents. *arXiv preprint arXiv:2308.03688*, 2023.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. *arXiv preprint arXiv:2402.17753*, 2024. ACL 2024.

Mark A. McDaniel and Gilles O. Einstein. *Prospective Memory: An Overview and Synthesis of an Emerging Field*. Sage Publications, 2007.

Avni Mittal. Did you forget what i asked? prospective memory failures in large language models. *arXiv preprint arXiv:2603.23530*, 2026. TrustNLP 2026.

Ali Modarressi, Ayyoob Imani, Mohsen Fayyaz, and Hinrich Schütze. Ret-LLM: Towards a general read-write memory for large language models. *arXiv preprint arXiv:2305.14322*, 2023.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive APIs. *arXiv preprint arXiv:2305.15334*, 2023.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. *arXiv preprint arXiv:2307.16789*, 2023.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: A temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025.

Peter G. Rendell and Fergus I. M. Craik. Virtual week and actual week: Age-related differences in prospective memory. *Applied Cognitive Psychology*, 14(7):S43–S62, 2000.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2023.

Rebekah E. Smith. The cost of remembering to remember in event-based prospective memory: Investigating the capacity demands of delayed intention performance. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 29(3):347–361, 2003.

Theodore Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas Griffiths. Cognitive architectures for language agents. *Transactions on Machine Learning Research*, 2024.

Zhen Tan, Junyan Li, Yuxin Liu, et al. In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. *arXiv preprint arXiv:2503.08026*, 2025. ACL 2025.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 2024.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. *arXiv preprint arXiv:2410.10813*, 2024.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for LLM agents. *arXiv preprint arXiv:2502.12110*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Tianhua Zhang, Xinjiang Wang, Q. Zhang, Qi Chen, Kun Li, Yaoqi Chen, Dingdong Wang, Helen Meng, and Yan Lu. TriggerBench: Investigating prospective memory for large language models. *arXiv preprint arXiv:2606.23459*, 2026.

Zeyu Zhang et al. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*, 2024.

others Zhao. AMA-Bench: Evaluating long-horizon memory for agentic applications. *arXiv preprint arXiv:2602.22769*, 2026.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19724–19731, 2024.

Wei Zhou, Xuanhe Zhou, Shaokun Han, Hongming Xu, Guoliang Li, Zhiyu Li, Feiyu Xiong, and Fan Wu. Are we ready for an agent-native memory system? *arXiv preprint arXiv:2606.24775*, 2026.