CS224V · AGENTIC AI · FALL 2026

Agent Context Management

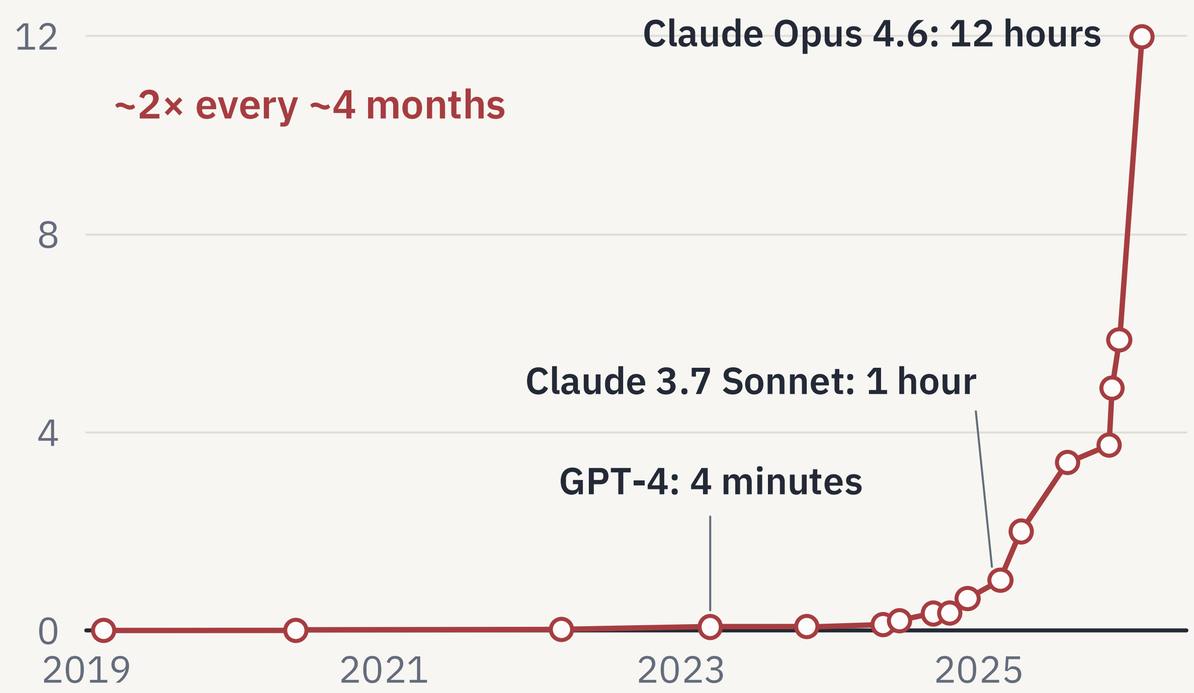
Yucheng Jiang, Monica Lam
Stanford OVAL



INTRODUCTION

Agents now run for hours

Task length agents finish, in hours of human work



2025 · Deep research

up to 600 tool calls

2026 · Software project

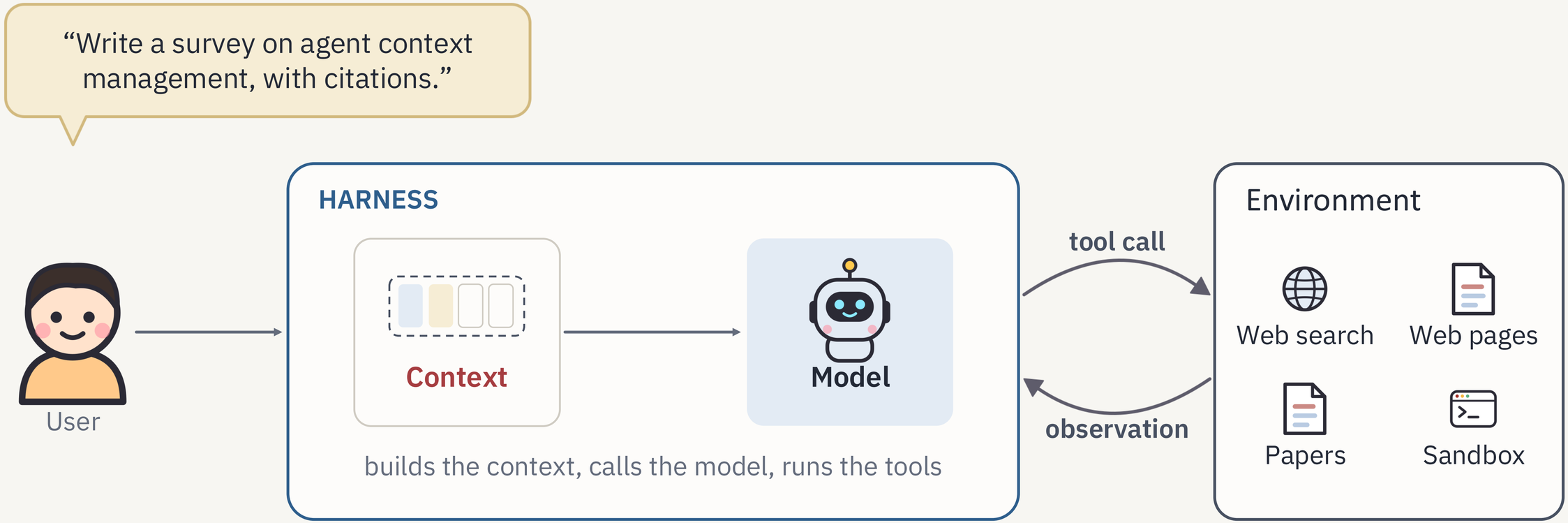
millions of tokens

2026 · Multi-week build

2 weeks, 16 agents

Kwa et al., “Measuring AI Ability to Complete Long Software Tasks,” NeurIPS 2025 · METR, “Time Horizon 1.1,” Jan 2026
MiroMind Team, “MiroThinker,” arXiv 2025 · Desai et al., “SWE-Marathon,” arXiv 2026
Anthropic, “Building a C compiler with a team of parallel Claudes,” Feb 2026

An agent: a model in a loop, run by a harness



Yao et al., “ReAct,” ICLR 2023 · Anthropic, “Building effective agents,” Dec 2024 · OpenAI, “openai/codex,” GitHub, Nov 2026

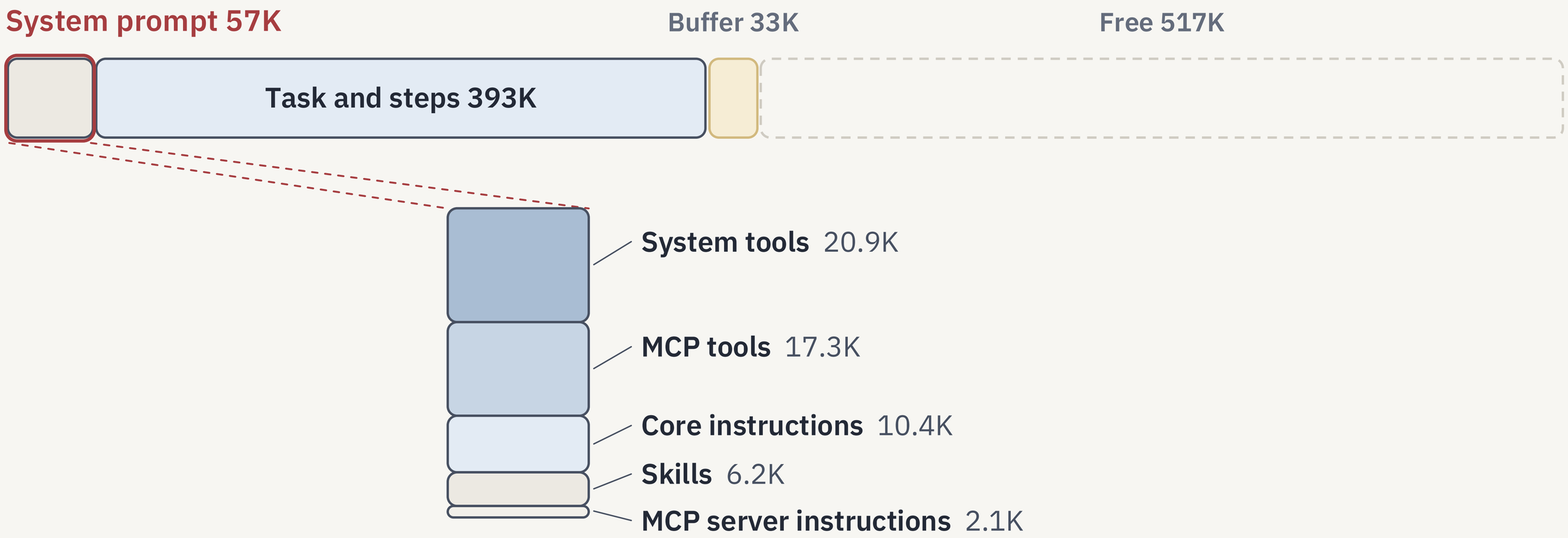
Steps and the trace

The model remembers nothing, so at every step the harness sends everything again.



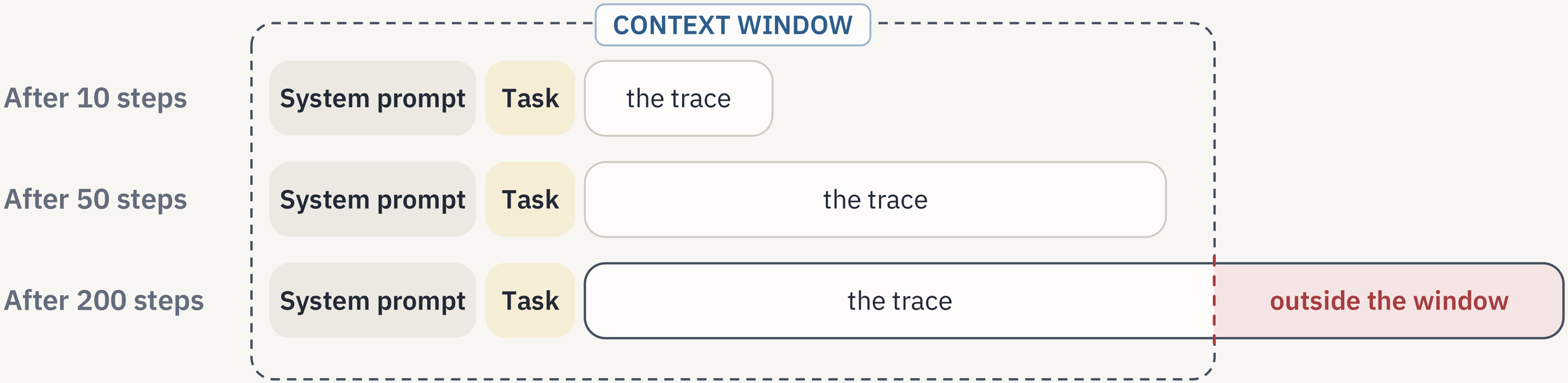
The context: everything the model sees in one call

One real Claude Code session: 450K of a 1M-token window in use.



Our measurement: Claude Code /context in one session, Nov 2026.

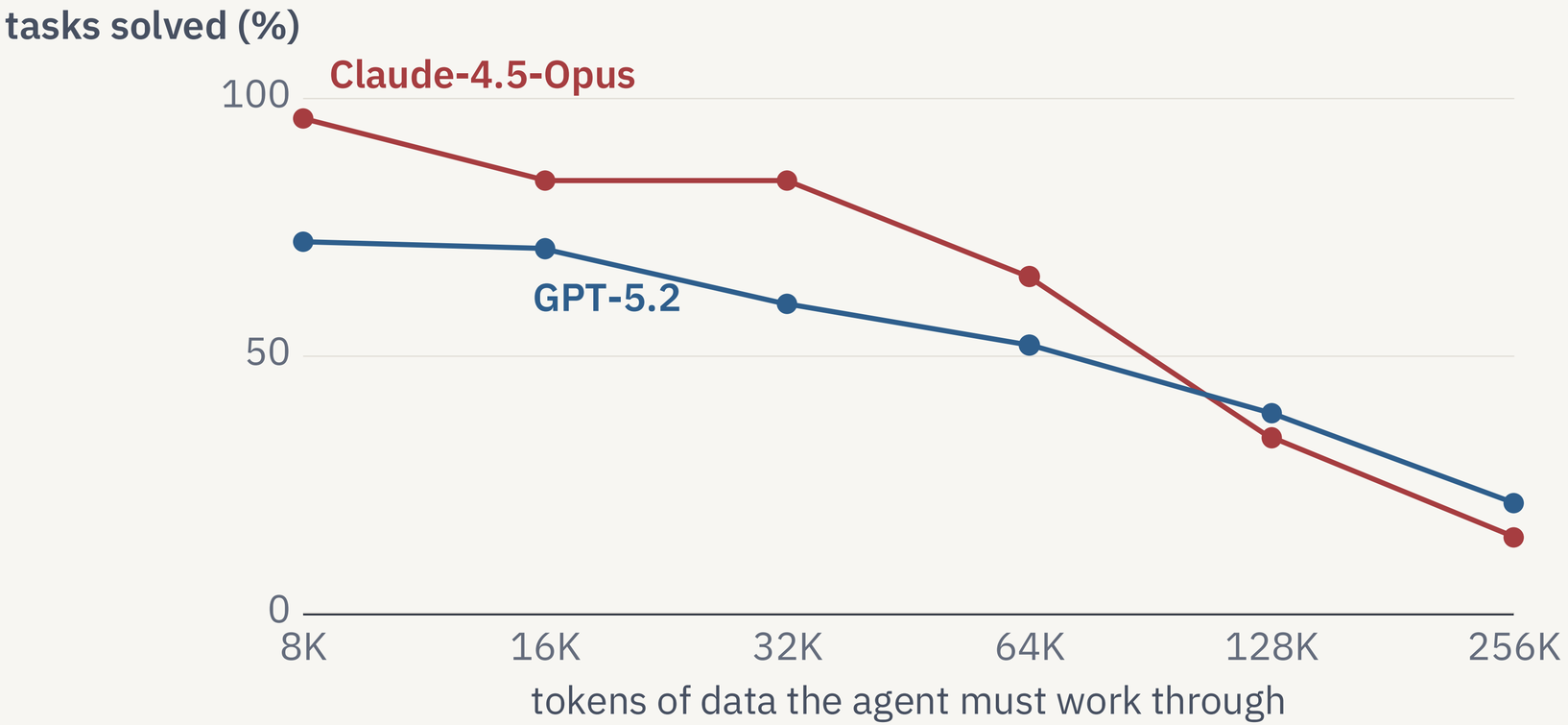
Problem 1: the trace outgrows the window



TAKEAWAY A policy must decide what goes into the window, and what to do with the rest.

Ye et al., “AgentFold,” ICLR 2026

Problem 2: more context, worse decisions



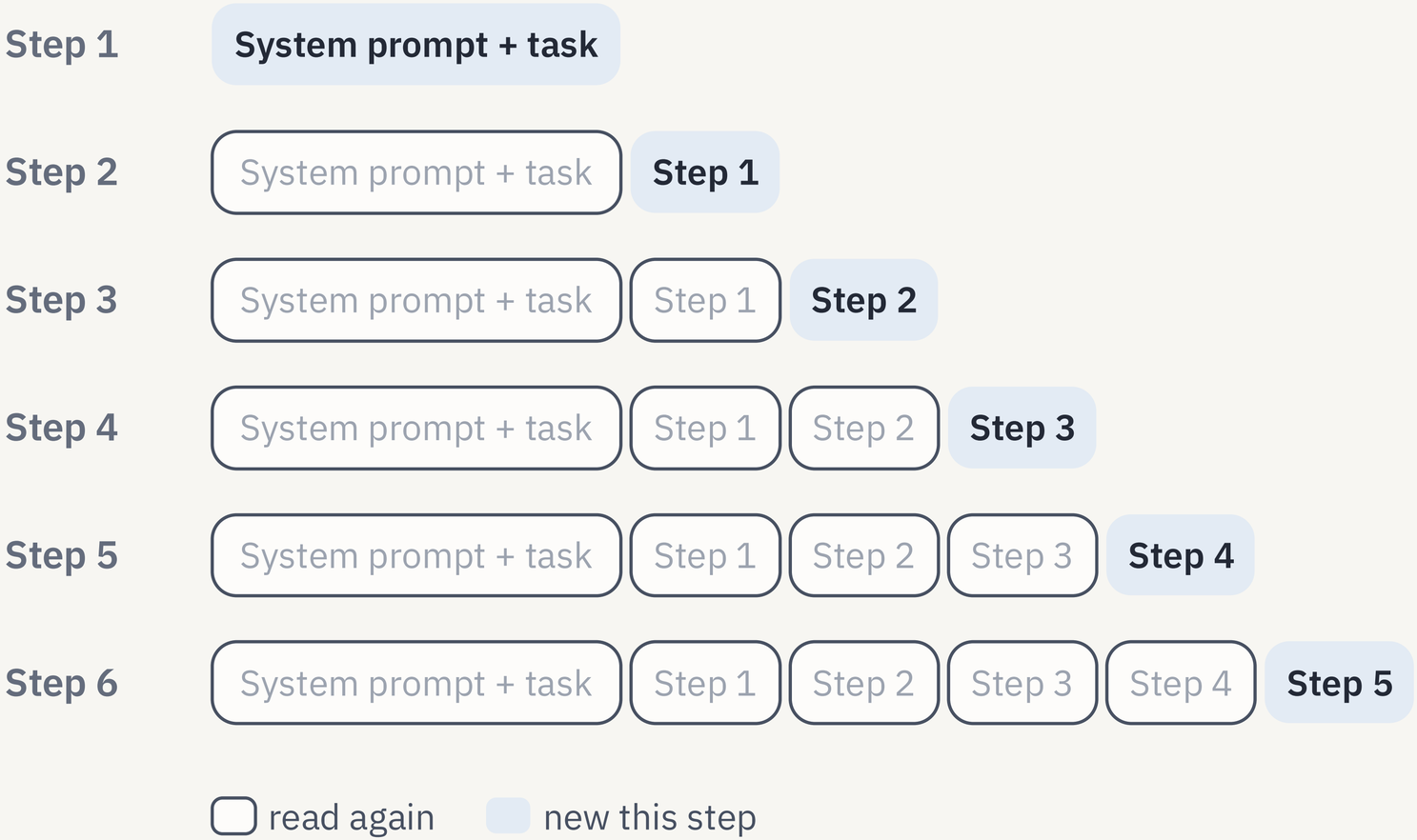
SAME TASKS

96% → 34%

Claude-4.5-Opus, as data grows from 8K to 128K tokens

Zeng et al., “LOCA-bench,” arXiv 2026 · Modarressi et al., “NoLiMa,” ICML 2025 · Liu et al., “Lost in the Middle,” TACL 2024

Problem 3: cost grows quadratically with task length

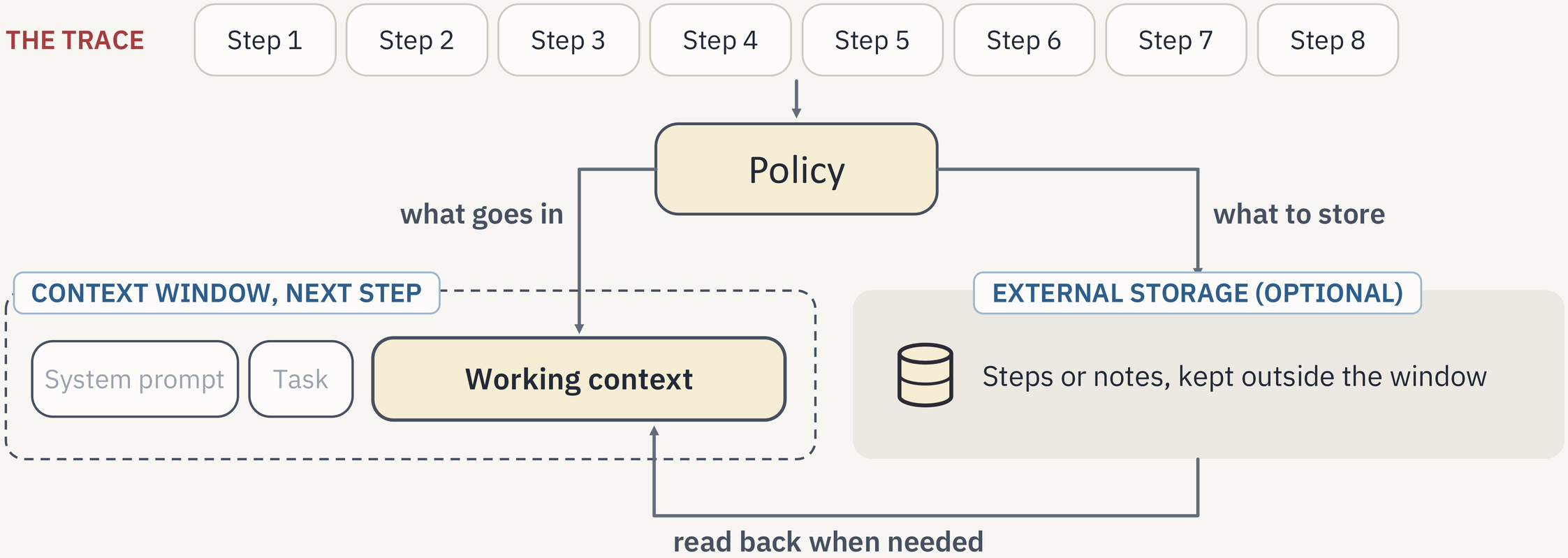


$O(T^2)$

tokens read over
a task of T steps

A context management policy

Before each step, decide what the model sees. Goal: finish the task within the window and budget.



INTRODUCTION

Today’s plan

1 **Context compression**

2 **What goes into the context window**

3 **External storage**

4 **Cross-session memory and skills**

PART 1

Context compression

Keep the working context small enough to fit, and good enough to act on.

PART 1 · CONTEXT COMPRESSION

Four ways to compress

THE HARNESS DECIDES

1 **Truncation**

2 **Summarization**

THE AGENT DECIDES

3 **Agent-managed context editing**

4 **Subtask Delegation**

1. Truncation: drop the oldest *K* steps

When the window is full, drop the oldest *K* steps.
The system prompt and the task are always kept.



PRO No extra model call; very simple

CON Dropped steps cannot be recovered

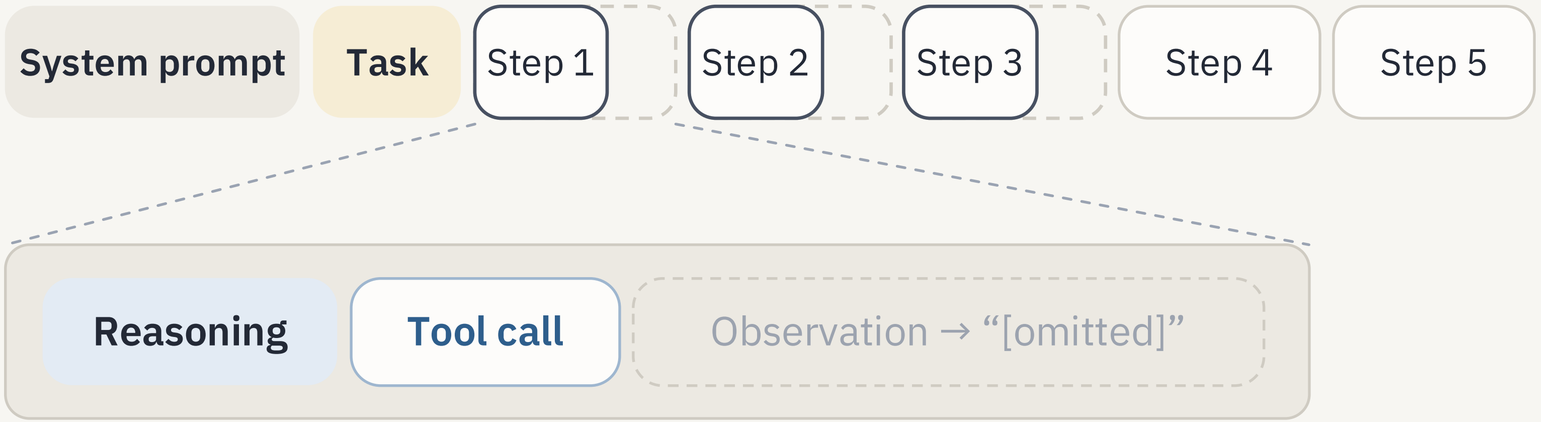
LangChain, “ConversationBufferWindowMemory,” 2022 · OpenAI, “Assistants API” (truncation_strategy), 2024

Two common truncation rules

Drop the *K* oldest steps



Observation masking



LangChain, “ConversationBufferWindowMemory,” 2022 · OpenAI, “Assistants API” (truncation_strategy), 2024
Yang et al., “SWE-agent,” NeurIPS 2024 · Anthropic, “Context editing,” Claude docs

Discussion: how should we choose *K*?



"What do we gain and lose with a very small *K*? With a very large *K*?"

$K = 1$

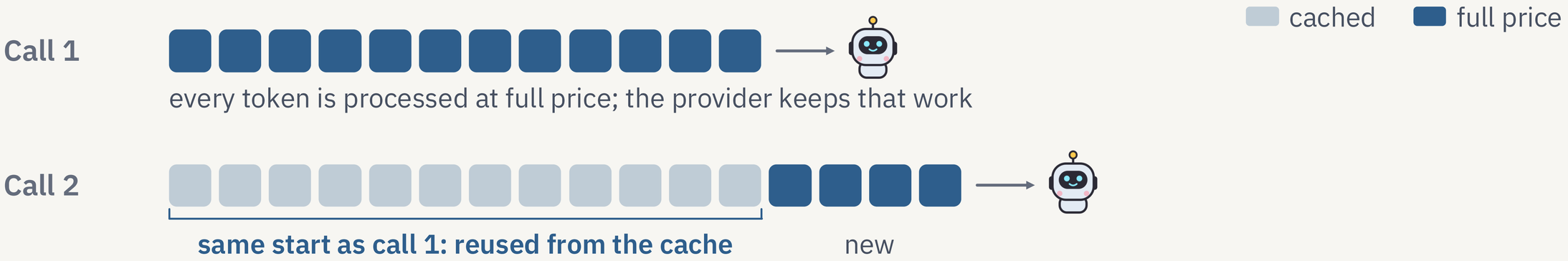
drop one step at a time

?

K = all old steps

drop everything at once

The prefix cache: cached tokens are cheaper



LIST PRICE PER 1M INPUT TOKENS (NOV 2026)

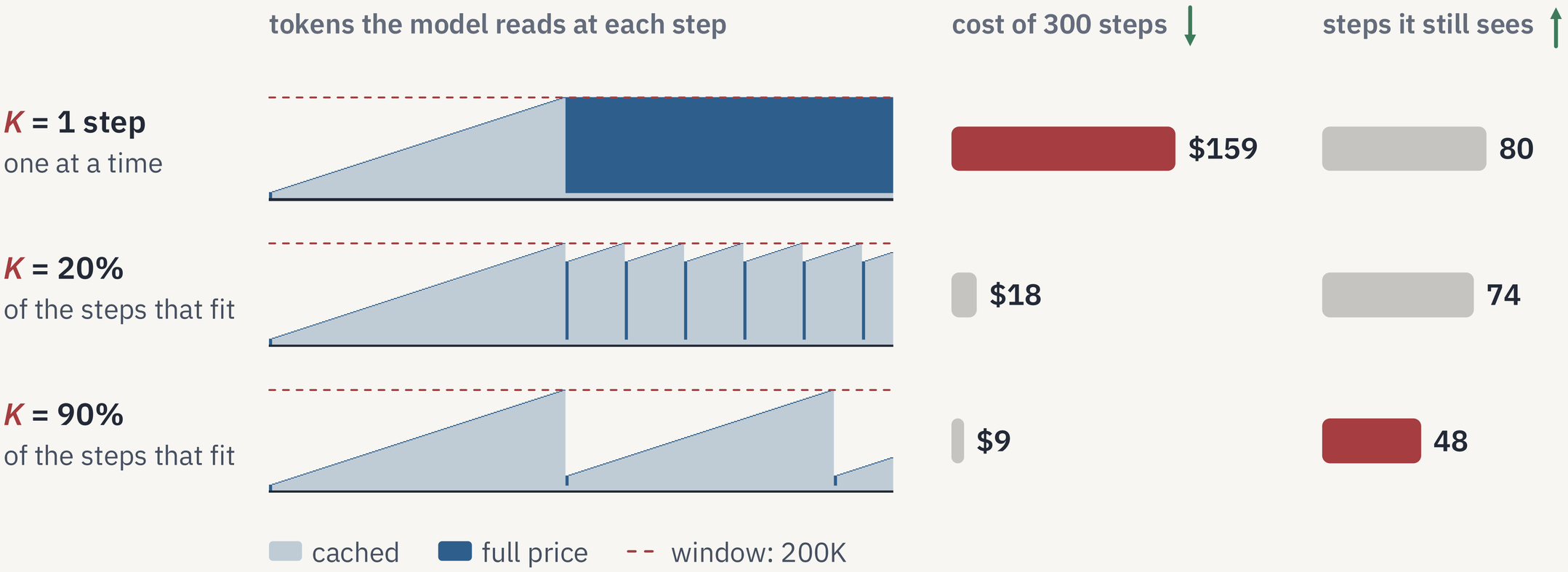
Model	Input	Cached input	
Claude Opus 5.5	$4.00	$0.20	**20× cheaper**
Claude Sonnet 5.5	$2.00	$0.10	**20× cheaper**
GPT-6 Astra	$10.00	$1.00	**10× cheaper**
GPT-6.1 Terra	$1.00	$0.05	**20× cheaper**

10–20×

cheaper to re-read a prefix that is still in the cache

Anthropic, “Pricing,” Claude docs · Anthropic, “Prompt caching,” Claude docs · OpenAI, “Pricing,” API docs

What *K* does to cost and to what the agent sees



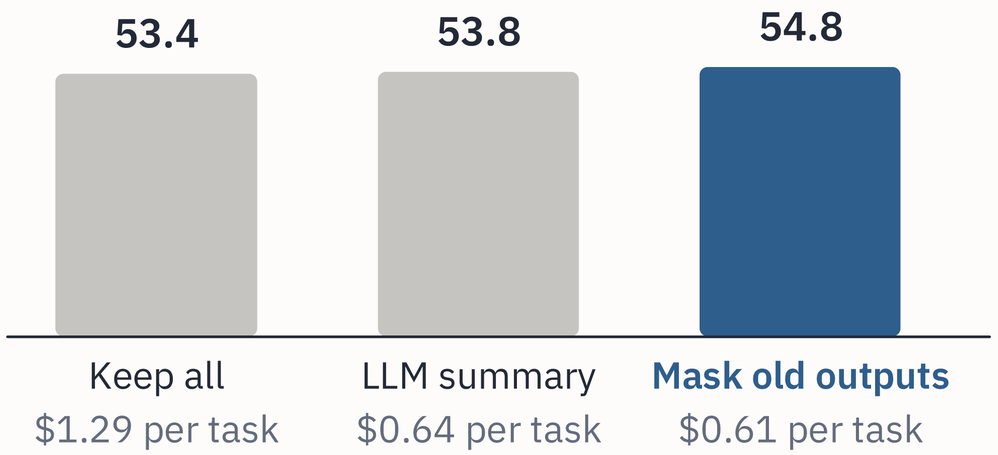
TAKEAWAY Small *K*: sees more past steps, but costs far more. Large *K*: cheap, but sees fewer.

Our simulation: 10K-token prompt and task, 2K tokens per step, 200K window (about 95 steps), 300 steps, Opus 5.5 prices.

Truncation is a strong baseline

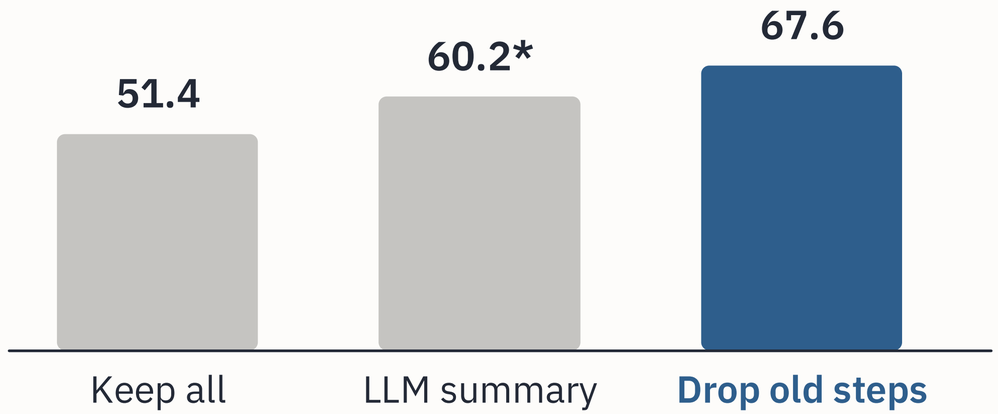
SWE-bench Verified

Qwen3-Coder 480B · tasks solved (%)



BrowseComp

DeepSeek-V3.2 · accuracy (%)



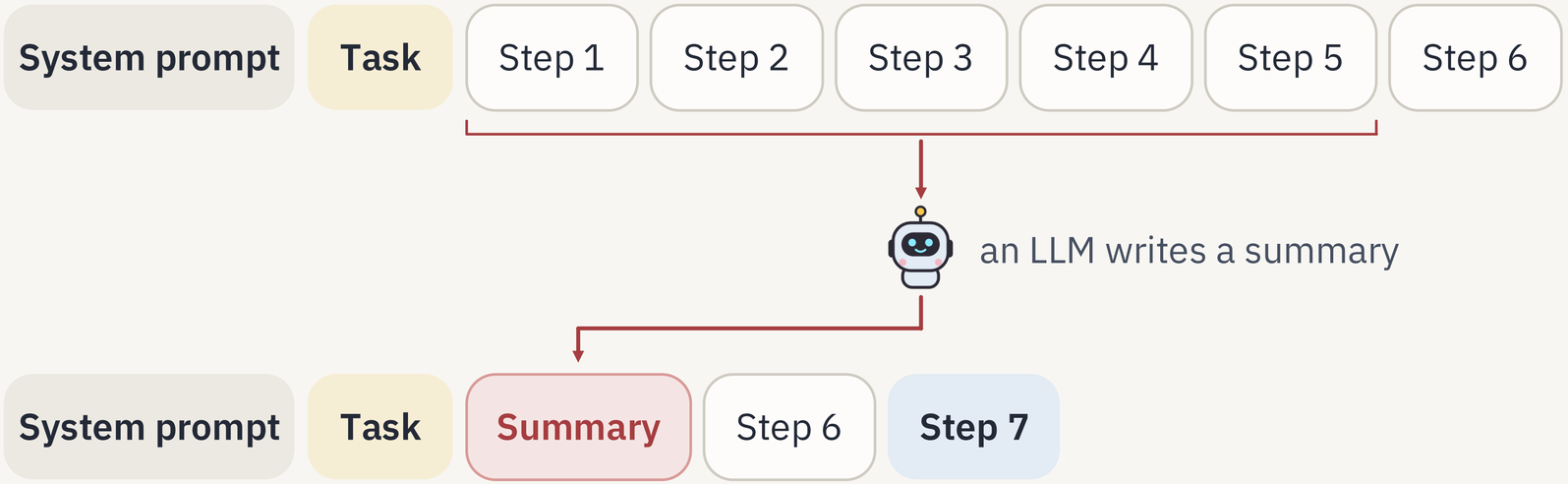
TAKEAWAY Most old tokens are tool results the agent has already acted on.

Lindenbauer et al., “The Complexity Trap,” NeurIPS 2025 Workshop · DeepSeek-AI, “DeepSeek-V3.2,” arXiv 2025

* best of several step budgets (“up to 60.2”)

2. Summarization: replace old steps with a summary

When the window is nearly full, an LLM summarizes the past. Also called compaction.



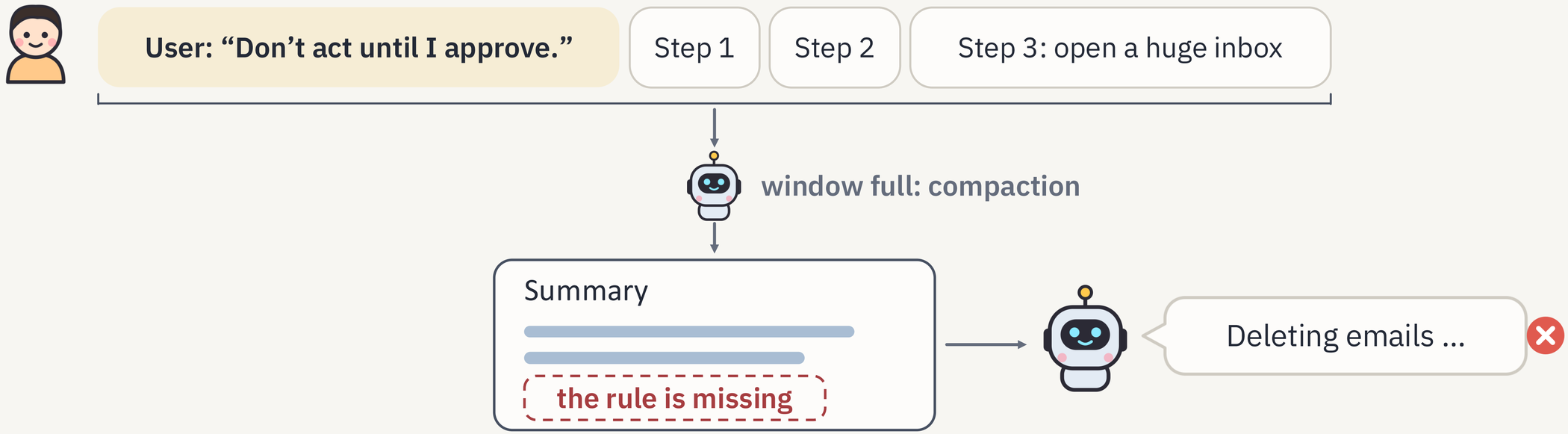
PRO Something of every old step survives

CON Extra model call; what it leaves out is lost

Packer et al., “MemGPT,” arXiv 2023
Wang et al., “Recursively Summarizing Enables Long-Term Dialogue Memory in Large Language Models,” Neurocomputing 2025
OpenHands, “Context Condensation for More Efficient AI Agents,” Apr 2025

Summaries are still lossy

A REAL INCIDENT

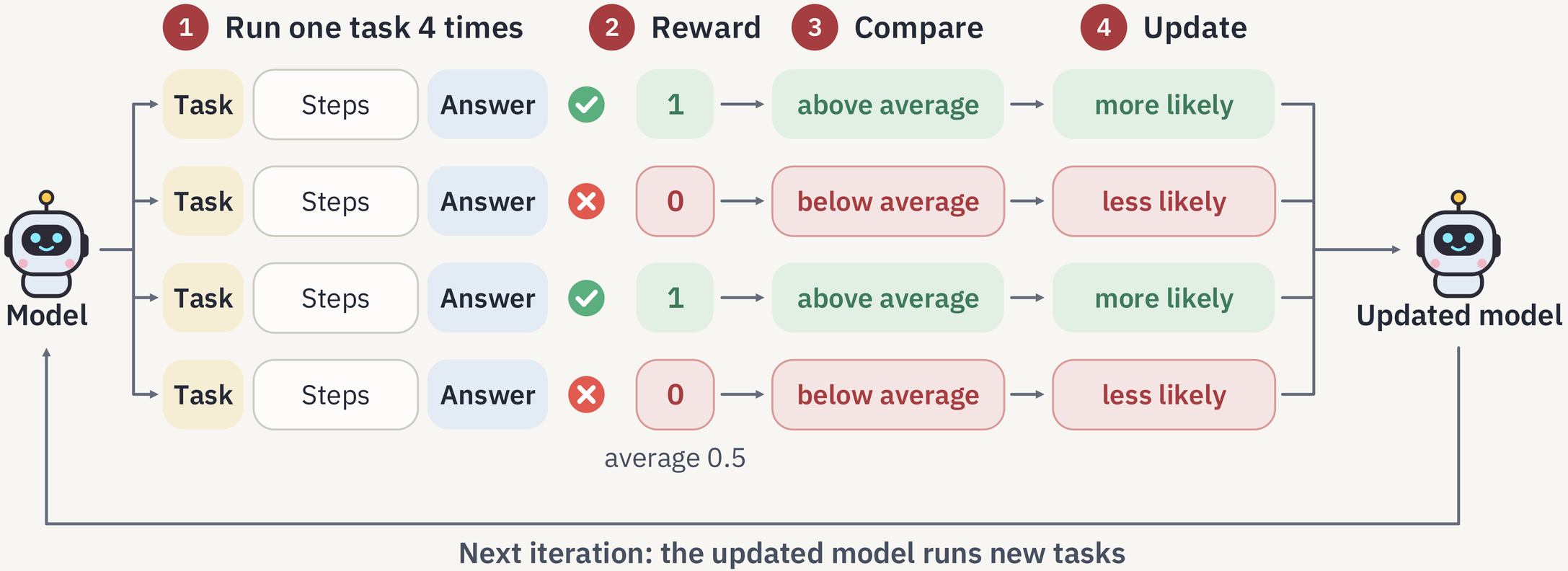


TAKEAWAY Compaction often drops user rules: one benchmark found only 17% kept, on average.

Simon Willison, “A quote from Summer Yue,” Feb 2026 · Wang et al., “Lost in Context Compaction,” arXiv 2026

Training with RL: GRPO in one picture

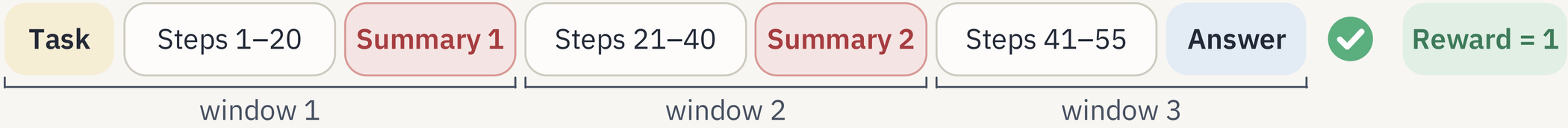
Can training make the summaries better? The usual tool is reinforcement learning (RL).



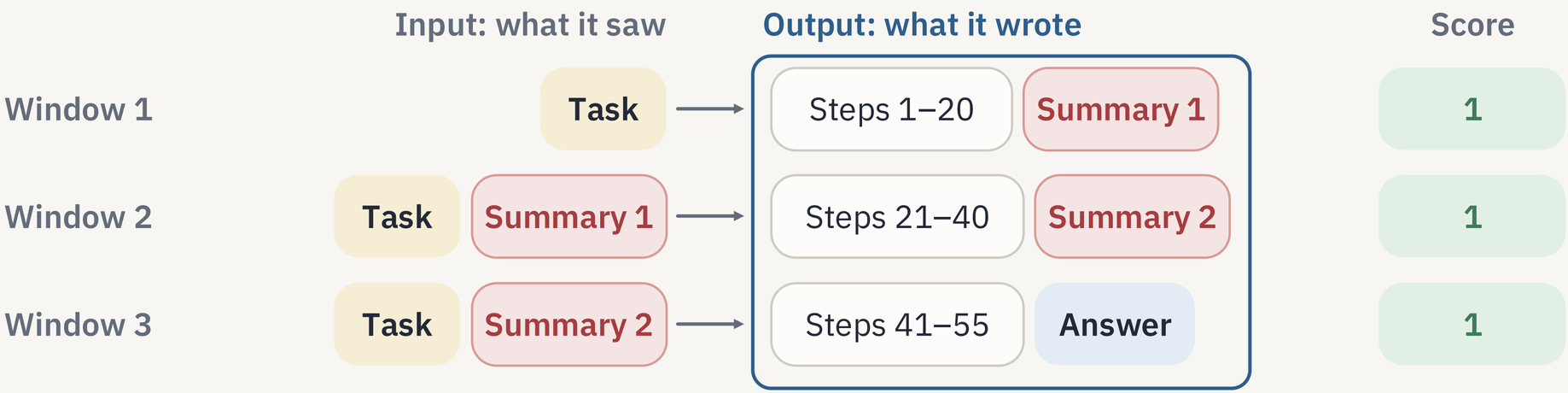
Shao et al., “DeepSeekMath” (GRPO), arXiv 2024

Training to summarize with GRPO

1 Roll out with summaries: one rollout spans 3 windows



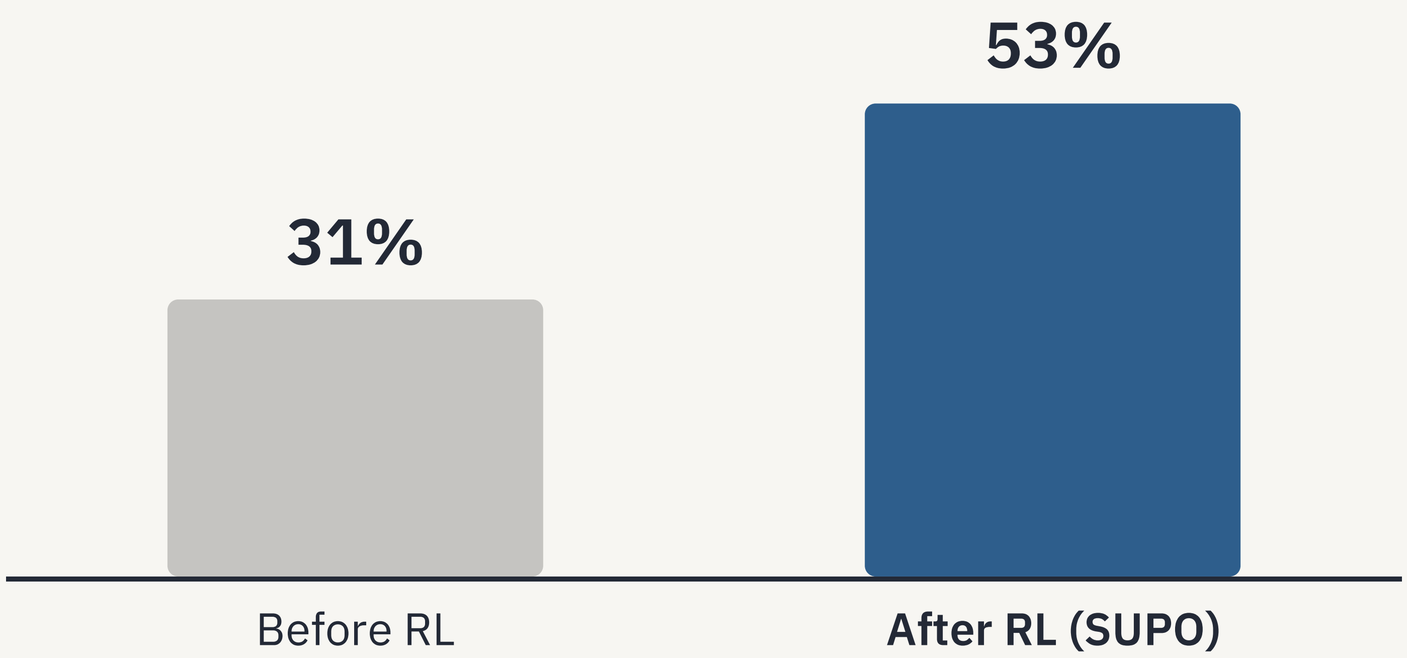
2 One training example per window, each with the rollout's score



Wu et al., "ReSum," arXiv 2025 · Lu et al., "Beyond the Context Window" (SUPO), ACL 2026 · Yu et al., "MemAgent," ICLR 2026
Cursor, "Training Composer for longer horizons," Mar 2026

Result: RL training lifts accuracy from 31% to 53%

Accuracy on BrowseComp-Plus. Same model, same test.



Lu et al., “Beyond the Context Window” (SUPO), ACL 2026

Reflection: a finer action space?

SO FAR: TWO ACTIONS, BOTH ON THE WHOLE OLD BLOCK

Drop it → gone

Summarize it → Summary

FINER ACTIONS, ONE STEP AT A TIME?

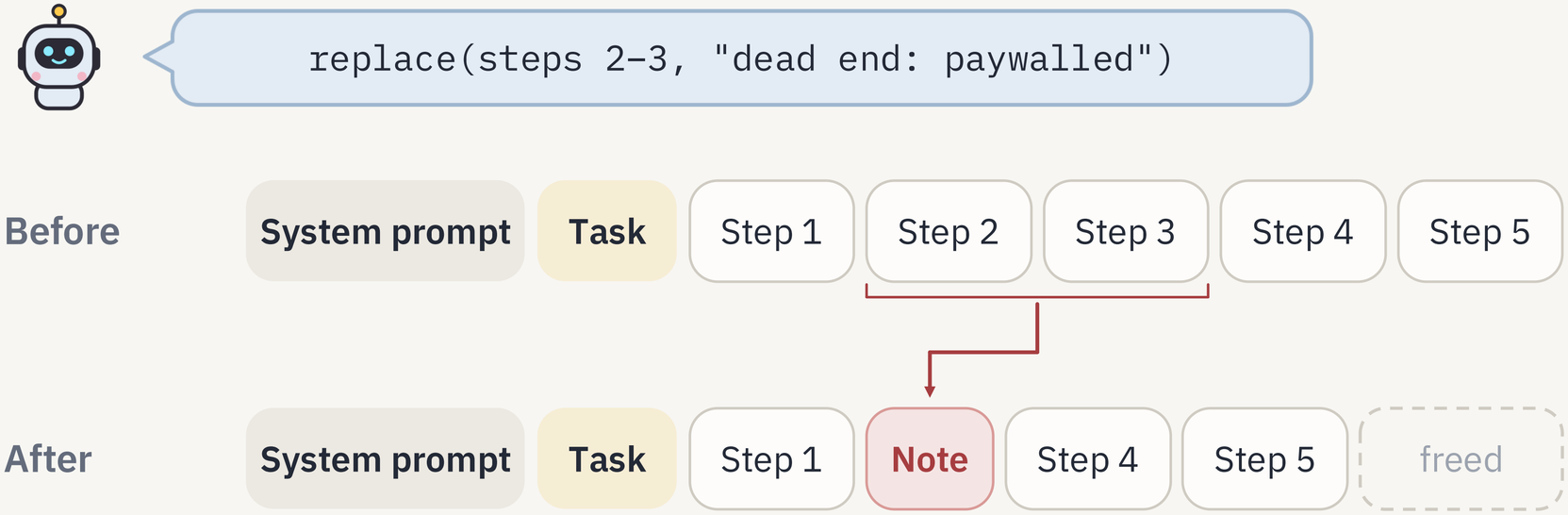
- drop one step
- trim one observation
- turn a dead end into a note
- keep a step as it is



“Could the agent pick from finer actions, step by step?”

3. Agent-managed context editing

Give the agent tools that edit its own context. It decides when to use them.



Zhang et al., “Memory as Action,” Findings of ACL 2026 · Lu et al., “LongSeeker,” arXiv 2026
Shao et al., “Context Language Models,” arXiv 2026

The action space: what can the agent edit?

EDIT	BEFORE → AFTER	USED BY
Summarize now		Self-Sum
Delete steps		MemAct, Agent-Omit
Replace steps with a note		AgentFold, LongSeeker
Trim an observation		LongSeeker
Edit with any code	 the context, as a file	Context Language Models

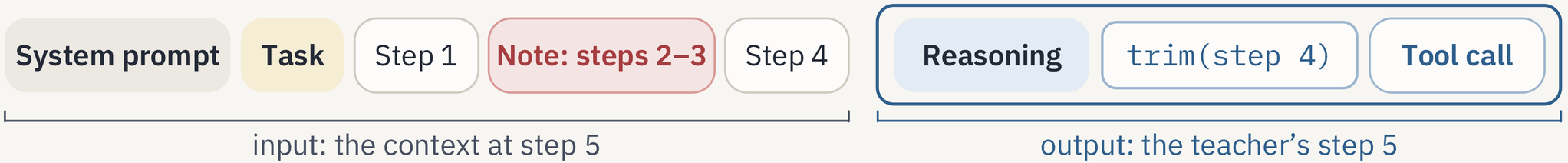
Wang et al., “Self-Sum,” Findings of ACL 2026 · Zhang et al., “Memory as Action,” Findings of ACL 2026
Ning et al., “Agent-Omit,” ICML 2026 · Ye et al., “AgentFold,” ICLR 2026 · Lu et al., “LongSeeker,” arXiv 2026
Shao et al., “Context Language Models,” arXiv 2026

Training editing agents (1): imitate a teacher

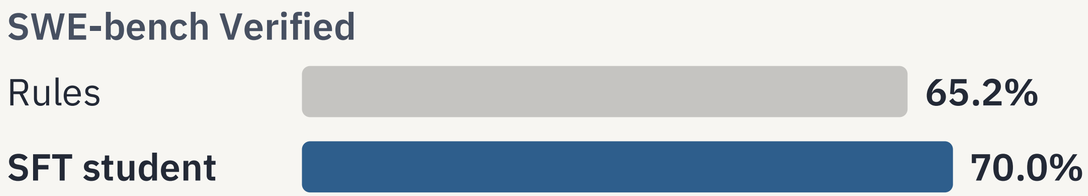
1 Collect traces from a strong teacher



2 Turn each step into one training example



3 Fine-tune the student to copy it (SFT)



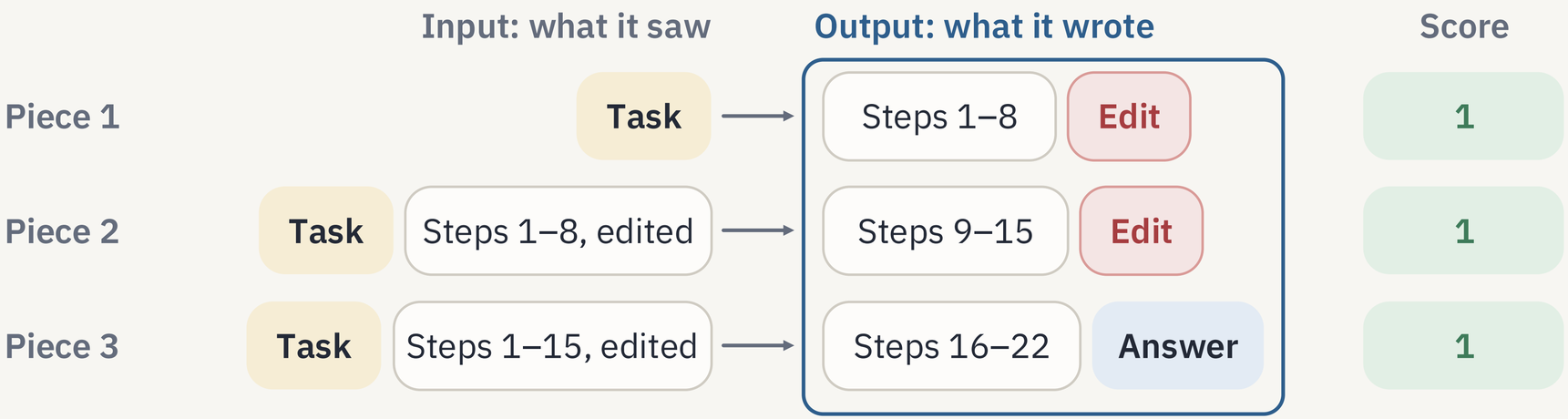
Deng et al., "Editing as Language," arXiv 2026 · Lu et al., "LongSeeker," arXiv 2026
Zhang et al., "Memory as Action," Findings of ACL 2026 · Wang et al., "Self-Sum," Findings of ACL 2026

Training editing agents (2): RL, as for summaries

1 **Roll out, then cut the rollout at each edit**



2 **One training example per piece, each with the rollout's score**



Zhang et al., "Memory as Action," Findings of ACL 2026

Training editing agents (3): reward design

1 The reward: task success, plus terms for editing well

Task outcome + **Efficiency bonus** + **Summary reward** − **Penalties**

Task outcome	Efficiency bonus	Summary reward	Penalties
	only if solved	only if solved	
1 or 0	for a smaller context	for good summaries	for going over budget

Only if solved: otherwise deleting everything earns the bonus, even when the task fails.

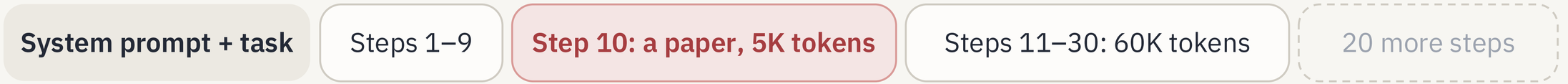
2 Does the reward design matter?



Ning et al., “Agent-Omit,” ICML 2026 · Wang et al., “Self-Sum,” Findings of ACL 2026
Zhang et al., “Memory as Action,” Findings of ACL 2026 · Sun et al., “Scaling Long-Horizon Agent via Context Folding,” ICML 2026

Smaller is not always cheaper

No longer relevant. Keep it or delete it?



Keep it — 5K re-read 20 times, from the cache — **$0.02**

Delete it — 60K after it re-read once, at full price — **$0.24**

Deleting breaks the cache for everything after it: worth it only for big items near the end.

Anthropic, “Pricing,” Claude docs · Anthropic, “Prompt caching,” Claude docs
Prices from slide 16 (Claude Opus 5.5): $4 per 1M input tokens, $0.20 cached. Illustrative sizes.

Agent-managed context editing: pros and cons

PROS

- Keeps exactly what matters
- Can act before the window is full
- Covers deleting, summarizing, and saving

CONS

- Needs a strong or trained model
- Every edit costs tokens and turns
- Deletion is permanent
- Early edits break the cache

RULE OF THUMB Edit recent steps freely. Move old content out instead of deleting it.

Shao et al., “Context Language Models,” arXiv 2026 · Deng et al., “Editing as Language,” arXiv 2026
Zhang et al., “Memory as Action,” Findings of ACL 2026

Next: let the task’s structure decide

Editing

Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8
edit?	*edit?*	*edit?*	*edit?*	*edit?*	*edit?*	*edit?*	*edit?*

a decision at every step, and each can go wrong

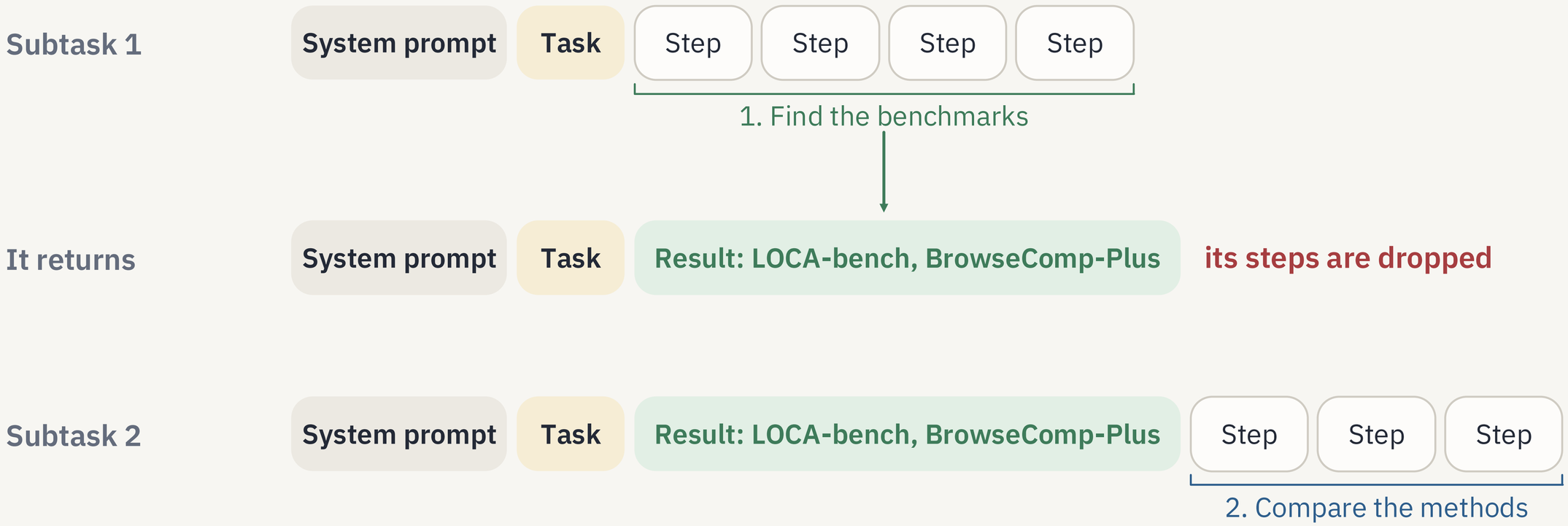
Idea



“Clean up once a subtask is done: keep its result, drop its steps.”

4. Subtasks: branch, then return

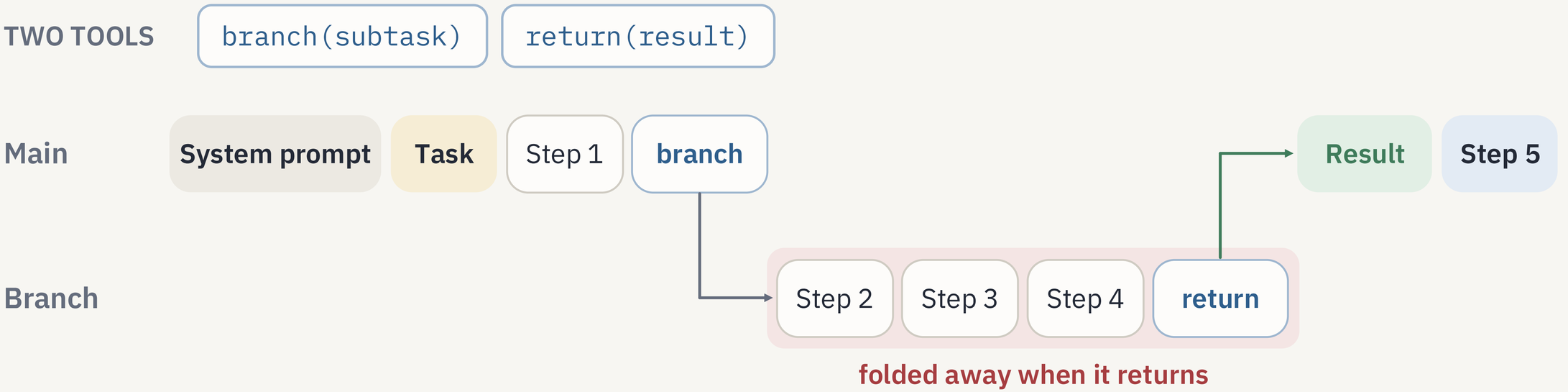
A long task splits into subtasks. When one is done, keep its result and drop its steps.



Hu et al., “HiAgent,” ACL 2025 · Sun et al., “Scaling Long-Horizon Agent via Context Folding,” ICML 2026

PART 1 · CONTEXT COMPRESSION

Context-Folding: branch and return as tools

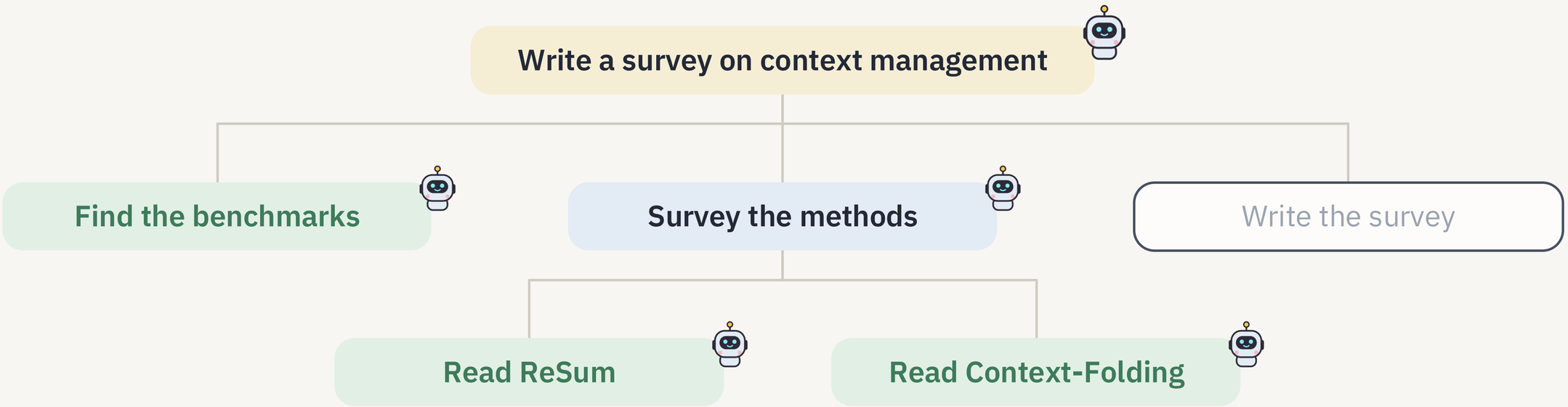


SAME MODEL, ALL TRAINED WITH RL · BROWSECOMP-PLUS



Sun et al., “Scaling Long-Horizon Agent via Context Folding,” ICML 2026

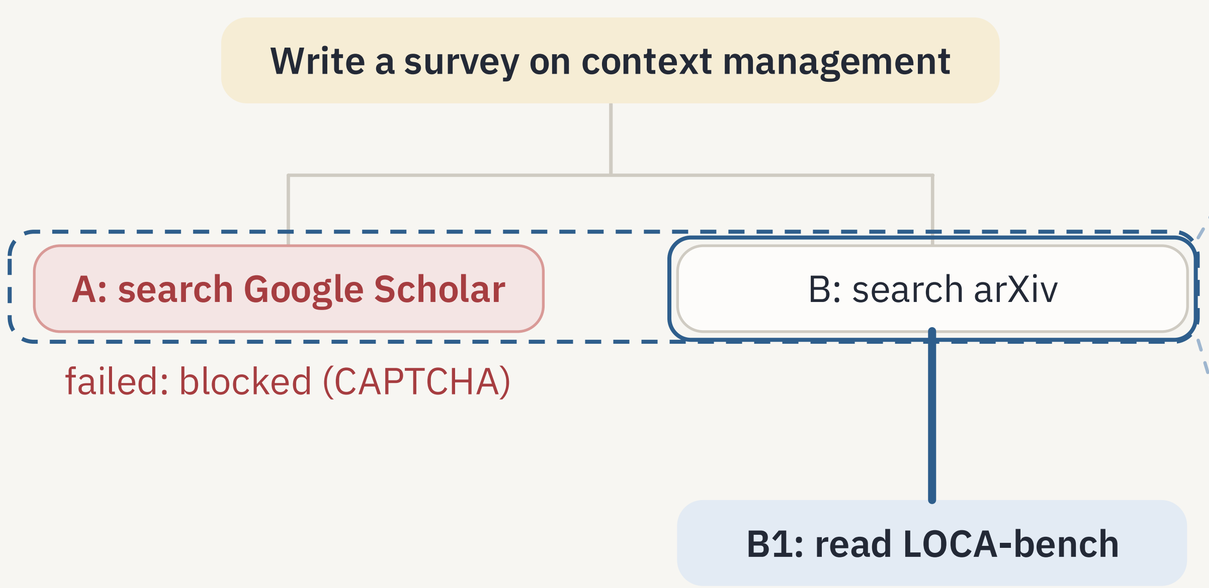
A branch is like delegating to a subagent



Each subtask runs in a fresh context; only its result comes back.

Anthropic, “How we built our multi-agent research system,” Jun 2025 · Anthropic, “Create custom subagents,” Claude Code docs
OpenAI, “Subagents,” Codex docs · Schroeder et al., “THREAD,” NAACL 2025

TreeFlex: the work as a tree of tries



INSIDE NODE B

Instruction

Find benchmarks for context management

Card: a summary of the work

Found: LOCA-bench, BrowseComp-Plus
Next: read how LOCA-bench is built

Restored when resumed

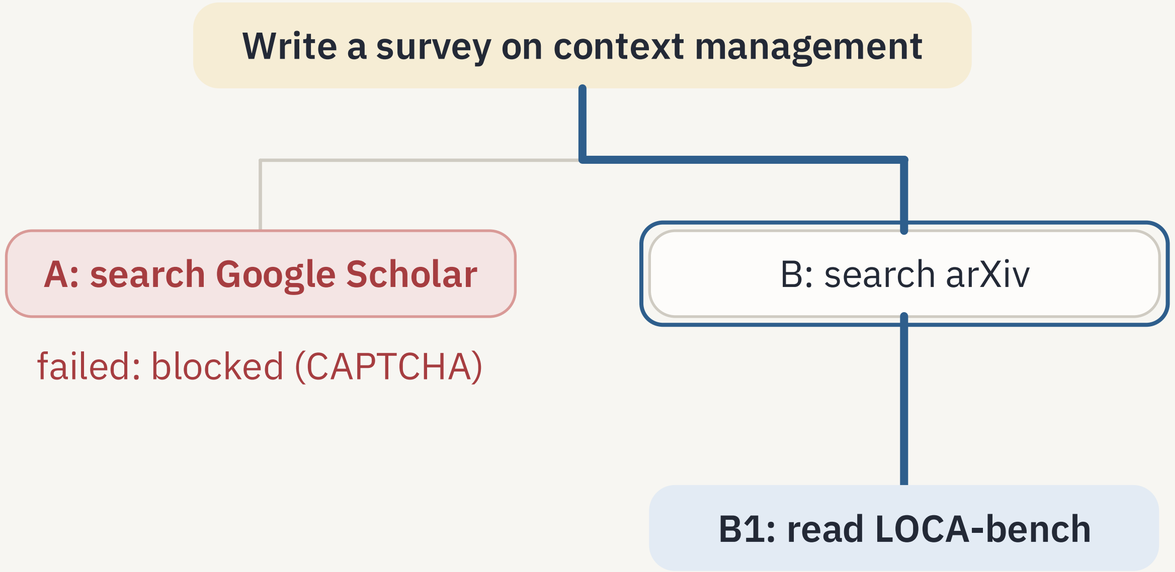
the cards on its path

Edge: the child continues from its parent's saved state

Siblings: different tries from the same point

Li et al., "TreeFlex," arXiv 2026
Illustrative example.

TreeFlex: work from a node, then save a new one



1 The planner picks a node

`navigate(B, “read LOCA-bench”)`

It sees every node’s card, A’s too.

2 Its path becomes the worker’s context

Task | **B’s card** | **Instruction**

Nothing from A.

3 The worker does the work

Task | B’s card | Instruction | **Step 1** | **…**

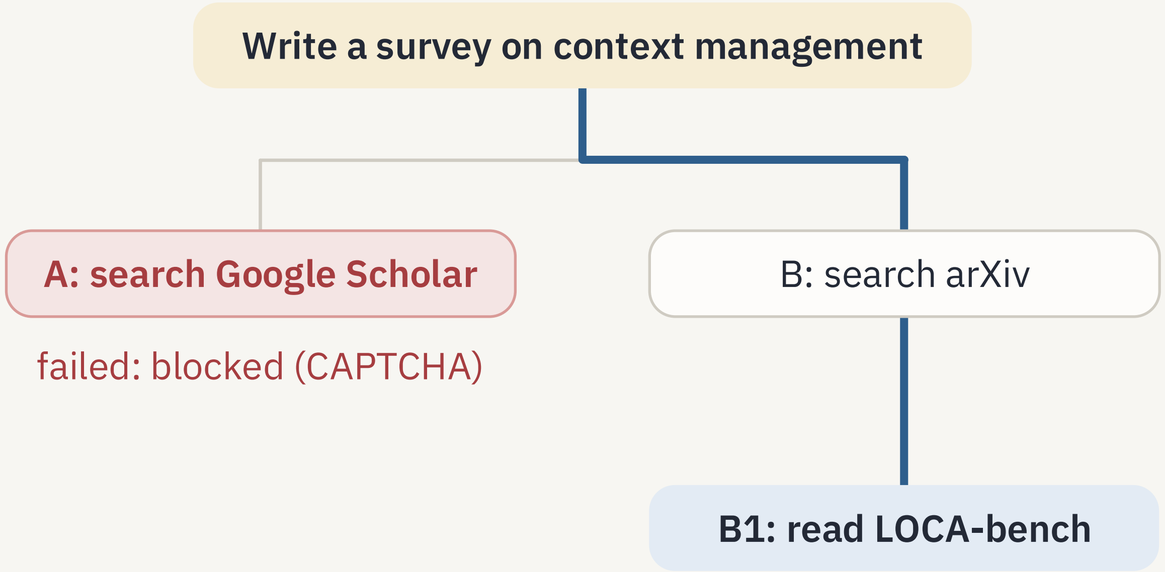
4 It saves a card: a new node, B1

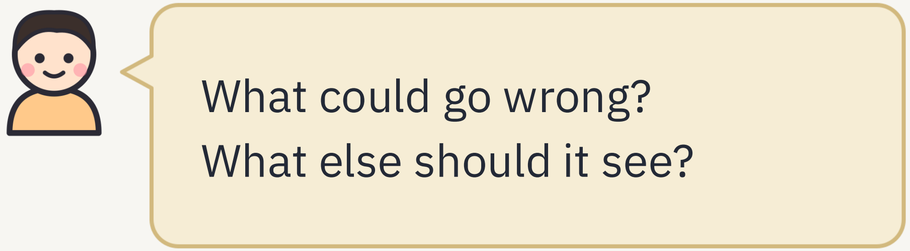
BrowseComp-Plus accuracy, same 4B model and 32K window



Li et al., “TreeFlex,” arXiv 2026
Illustrative example.

Discussion: what is missing from the path?

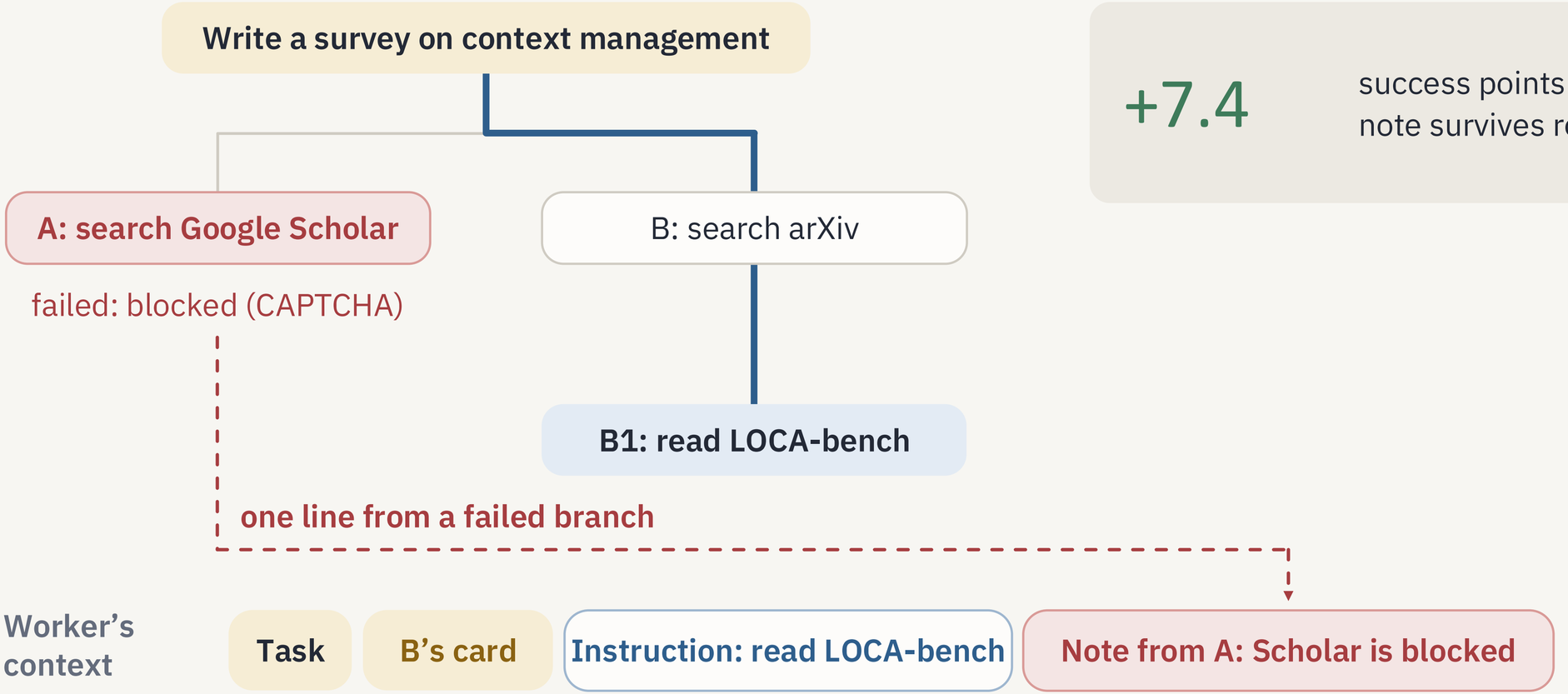




Worker's context



Better: the path, plus notes from other branches



+7.4 success points when a failure note survives rollbacks

Li et al., "TreeFlex," arXiv 2026 · Lu et al., "LongSeeker," arXiv 2026
Yu et al., "Rollback the World, Keep the Reflection," arXiv 2026

Context compression at a glance

	Who decides	Extra cost	Prefix cache	What can be lost	Training helps
Truncation	Harness	None	**Resets**	Old steps	—
Summarization	Harness	Summary calls	**Resets**	Details	**Yes**
Agent-managed context editing	Agent	Edit steps	Partly kept	What it deletes	**Yes**
Subtasks	Agent	Branch steps	**Kept**	Subtask steps	**Yes**

PART 2

What goes into the context window

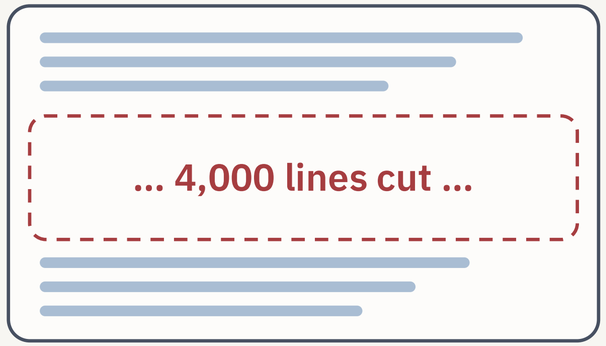
Keep each step from adding more than the agent needs.

Limit what each tool call returns

The cheapest place to manage context is before a result enters it.

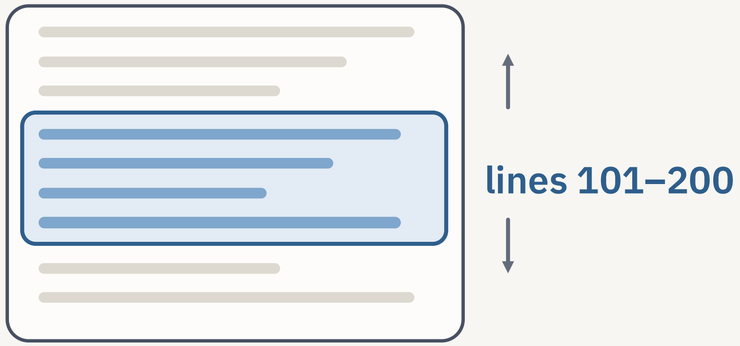
Keep head and tail

Codex: up to 10K tokens



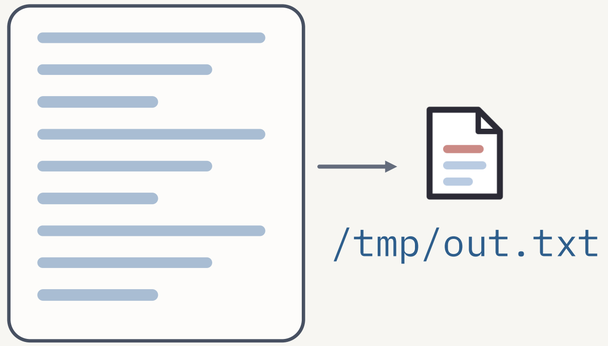
Show a window

SWE-agent: 100 lines



Save to a file

Claude Code: long output



the context gets the path
and a preview

Yang et al., “SWE-agent,” NeurIPS 2024 · OpenAI, “openai/codex,” GitHub, Nov 2026
Anthropic, “Tools reference,” Claude Code docs · Tran, “Where the Tokens Go,” ACL 2026 Workshop

One tool call per step fills the window

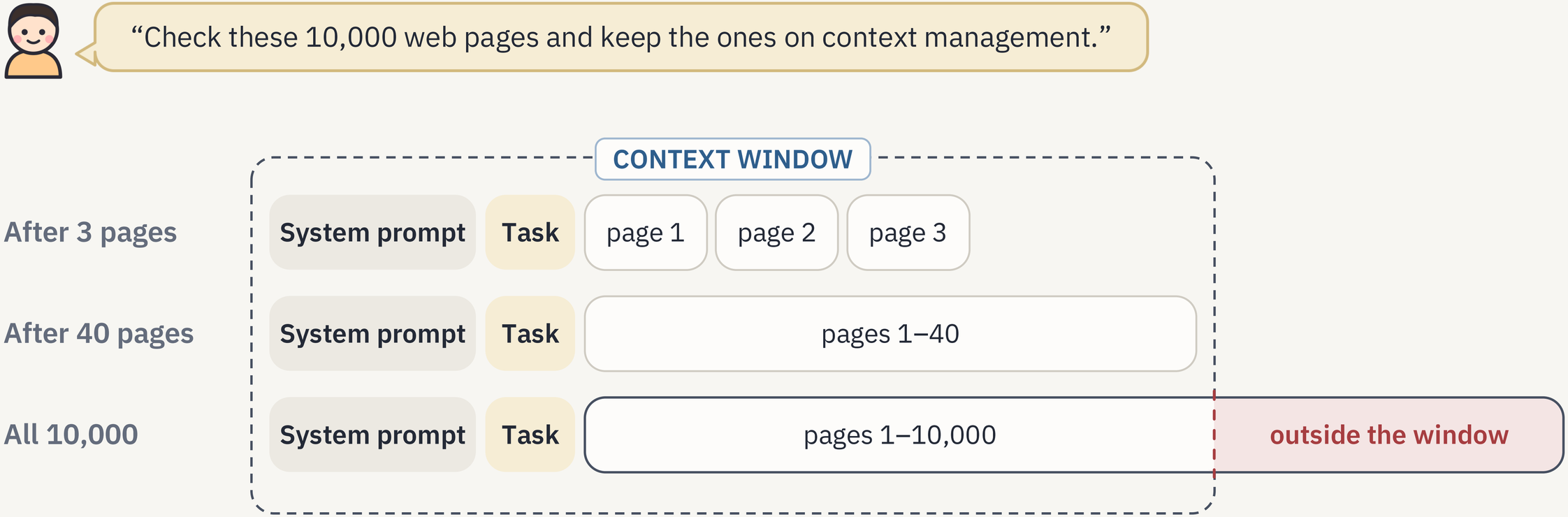


Illustration: about 5K tokens per page, a 200K window.

Programmatic tool calling: one program instead



“Check these 10,000 web pages and keep the ones on context management.”

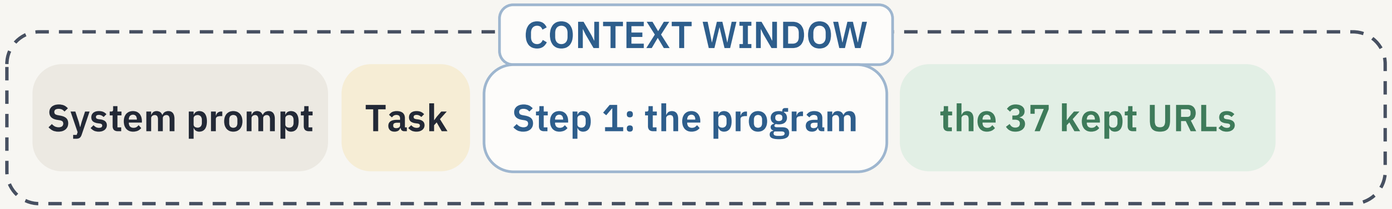


```
kept = []
for url in urls:
    page = fetch(url)
    if relevant(page):
        kept.append(url)
print(kept)
```

→ **the 37 kept URLs**

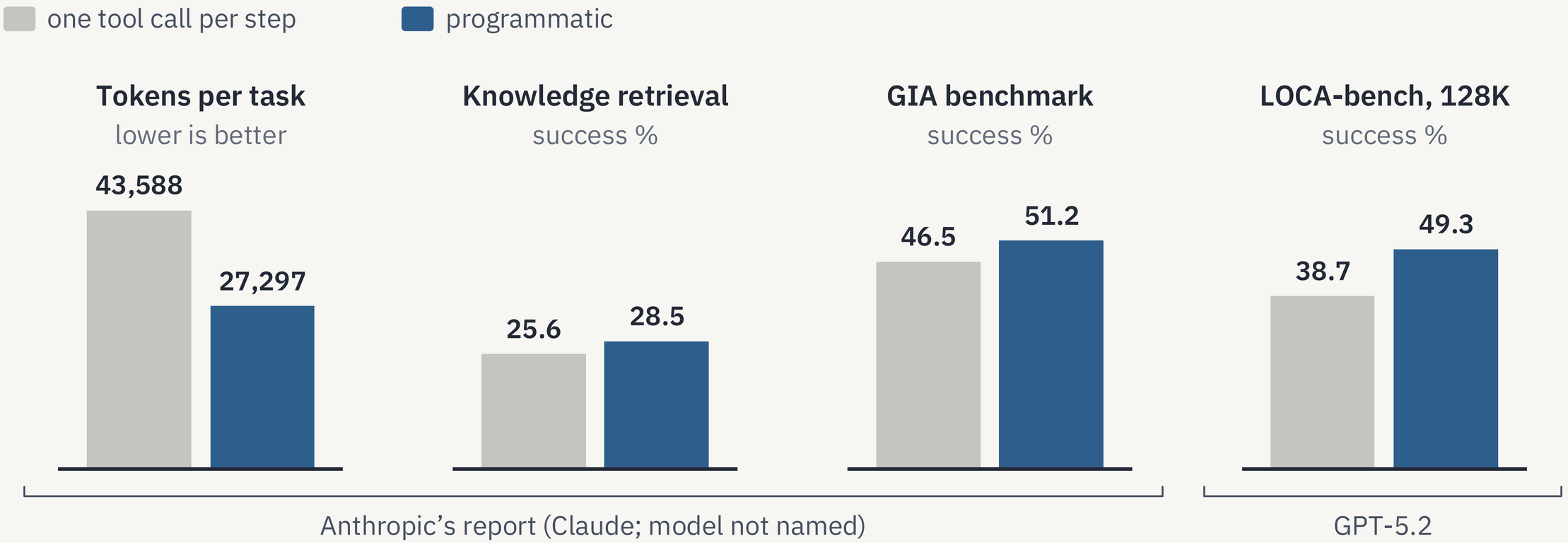
the only output

After the program



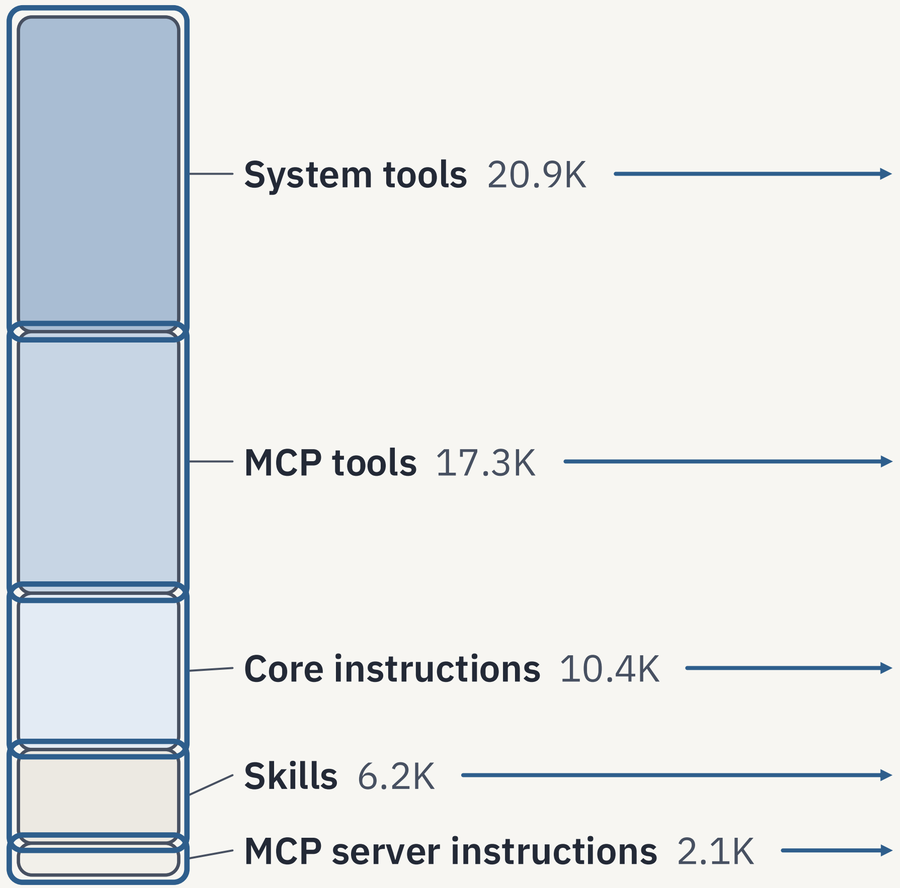
Wang et al., “Executable Code Actions Elicit Better LLM Agents” (CodeAct), ICML 2024
Anthropic, “Introducing advanced tool use on the Claude Developer Platform,” Nov 2025
OpenAI, “Programmatic Tool Calling,” API docs

Does programmatic tool calling help?



Anthropic, “Introducing advanced tool use on the Claude Developer Platform,” Nov 2025 · Zeng et al., “LOCA-bench,” arXiv 2026

Inside Claude Code’s system prompt



MCP server instructions 2.1K tokens

`claude-in-chrome`

“IMPORTANT: If the Chrome browser tools are deferred (must be loaded via ToolSearch before use), load them with ToolSearch before calling them, and batch every tool you expect to need into ONE ToolSearch call …”

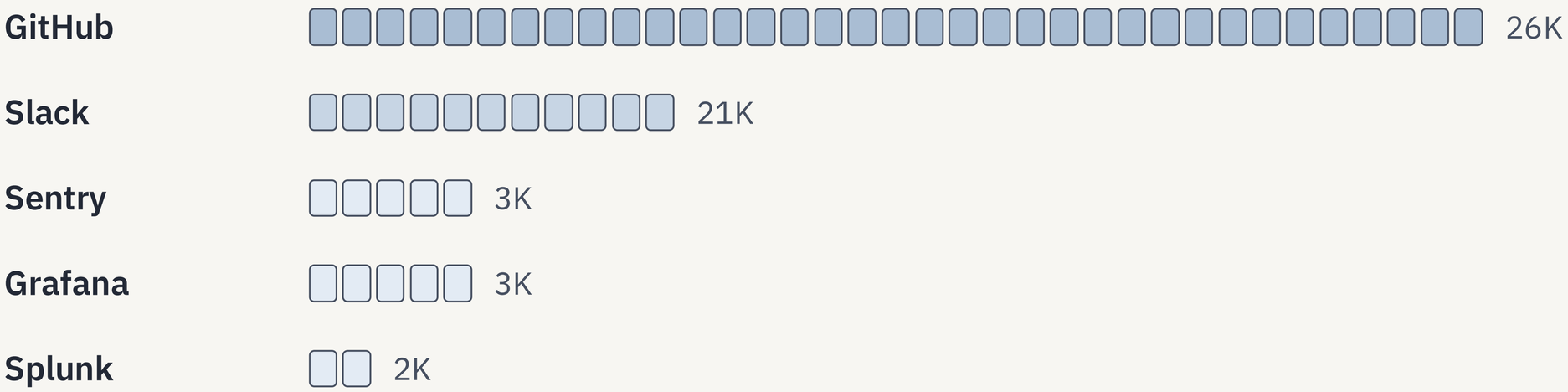
`computer-use`

“The app_* tools act on one granted app, the one a call names, and do not take the screen from the user.”

Examples: copied from a Claude Code session in the Claude desktop app, Nov 2026.
Token counts: the /context session in the introduction.

Tool definitions take up the window too

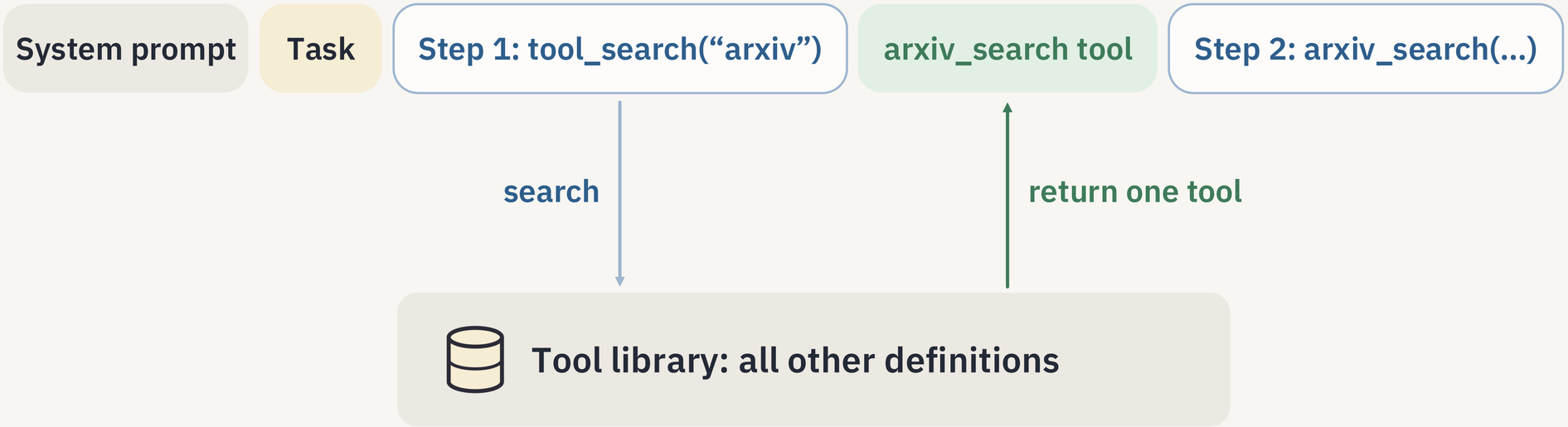
▯ = one tool definition, sent at every step



≈ 55K tokens of tool definitions

Anthropic, “Introducing advanced tool use on the Claude Developer Platform,” Nov 2025

Tool search: load definitions only when needed



Step 2 calls the tool it just found. Its definition is added at the end, so the cached prefix survives.

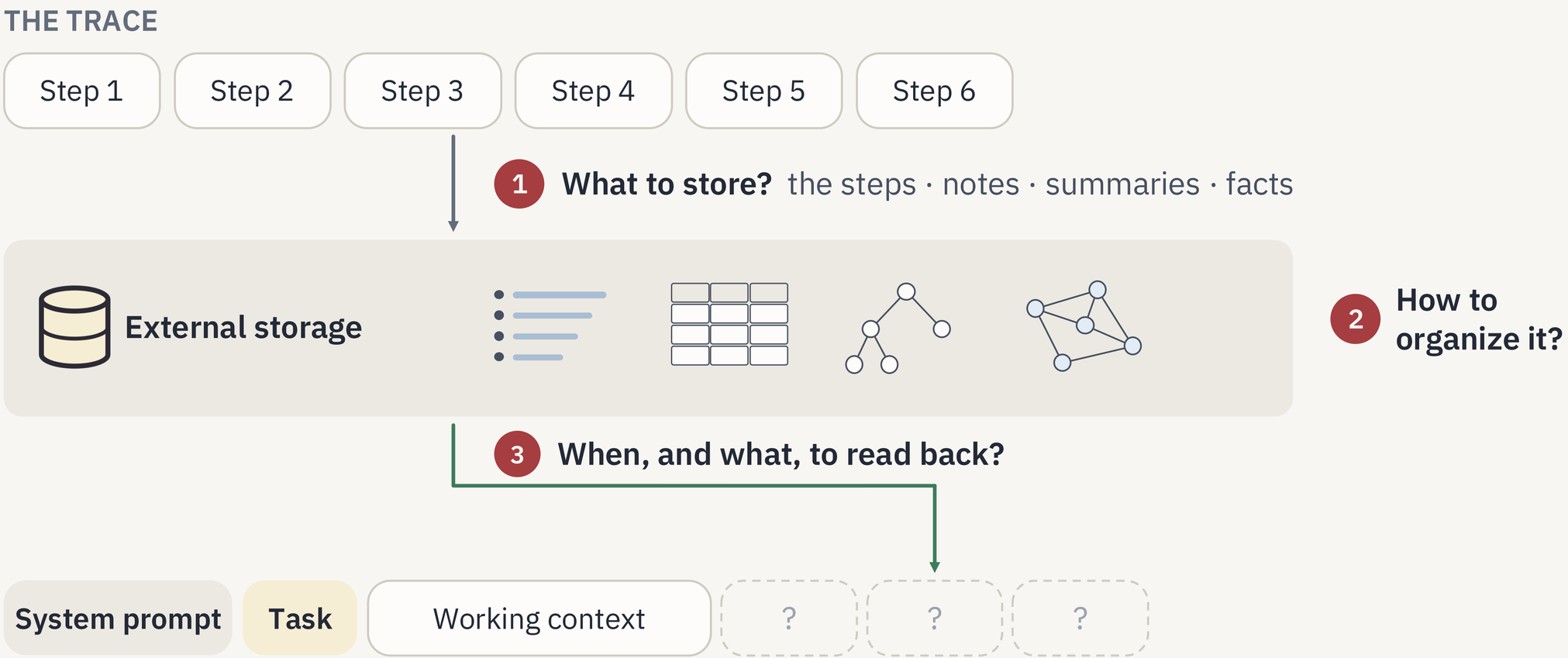
Anthropic, “Introducing advanced tool use on the Claude Developer Platform,” Nov 2025
Anthropic, “Connect Claude Code to tools via MCP,” Claude Code docs

PART 3

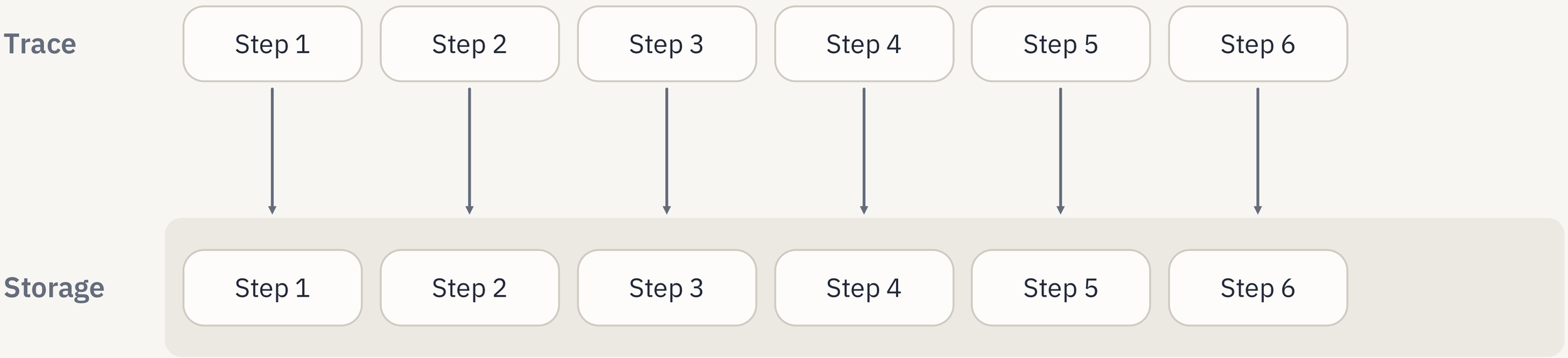
External storage

Keep what leaves the window outside it, and read it back when it is needed.

External storage: three design questions



What to store (1): the steps themselves

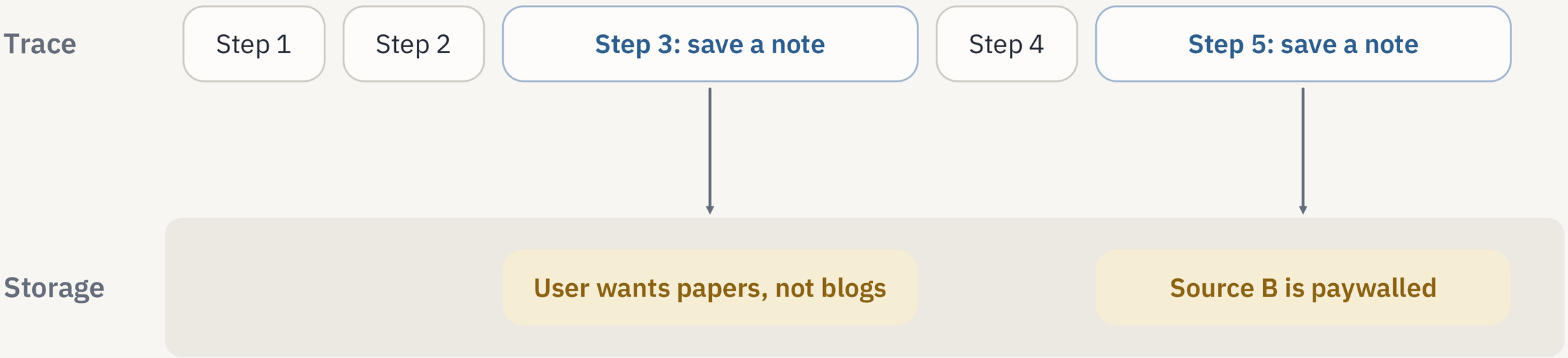


PRO Nothing is lost; any step can be read back

CON The right step is hard to find later

Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” NeurIPS 2020
Packer et al., “MemGPT,” arXiv 2023 · Lin et al., “Context as an Environment” (Scroll), arXiv 2026

What to store (2): notes the agent writes

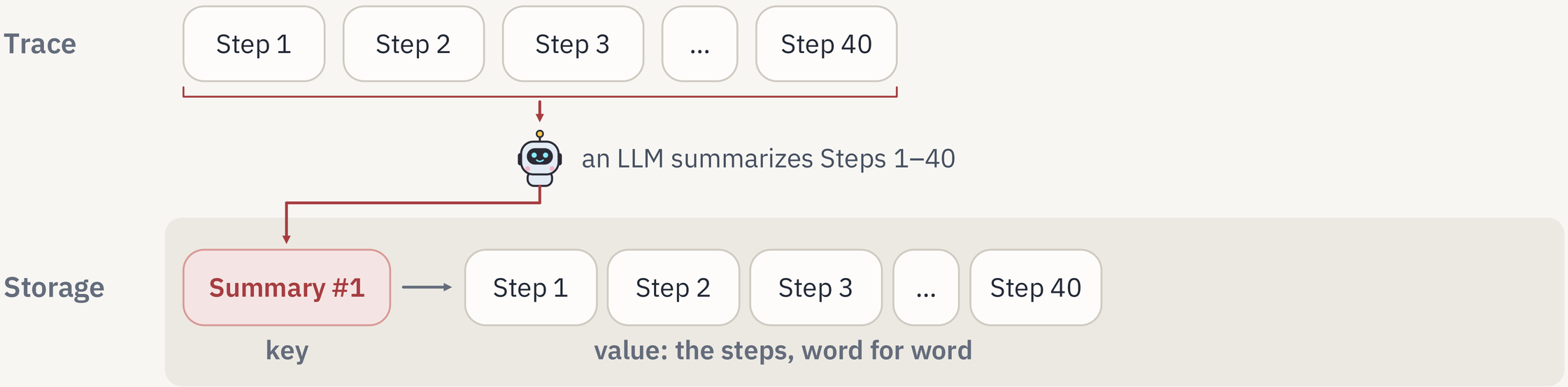


PRO Short, and written for later use

CON Must guess what will matter later

Packer et al., “MemGPT,” arXiv 2023 · Anthropic, “Memory tool,” Claude docs

What to store (3): the steps behind a summary



PRO Short context, and nothing is lost

CON Generating a good summary is hard

Li et al., “ACM,” arXiv 2026 · Wang et al., “Memex(RL),” arXiv 2026

What to store (4): facts extracted from the steps



PRO Small, and each fact can be updated

CON Extractor must guess what matters later

Chhikara et al., “Mem0,” ECAI 2025 · Rasmussen et al., “Zep,” arXiv 2025

How to organize it (1): a list

Trace

Step 11

Step 12: “The deadline moved to Nov 24.”

Storage

Topic: context management

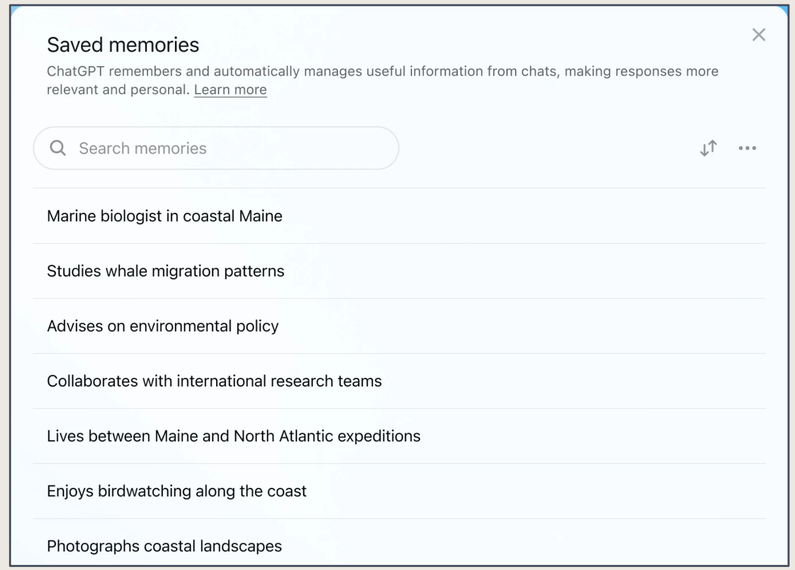
Cite papers, not blogs

~~Survey due Nov 20~~

UPDATE

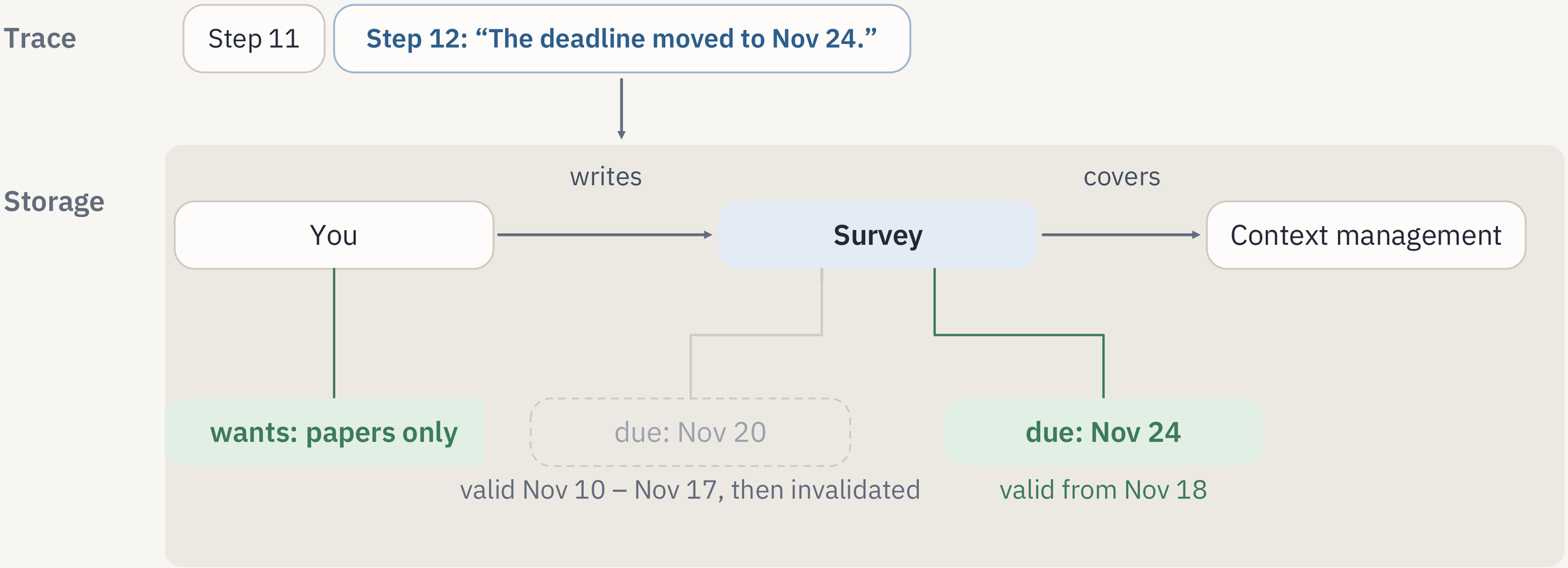
Survey due Nov 24

CHATGPT: SAVED MEMORIES



Chhikara et al., “Mem0,” ECAI 2025
Screenshot: OpenAI, “Dreaming,” Jun 2026

How to organize it (2): a graph



Rasmussen et al., “Zep,” arXiv 2025

How to organize it (3): a table

Trace

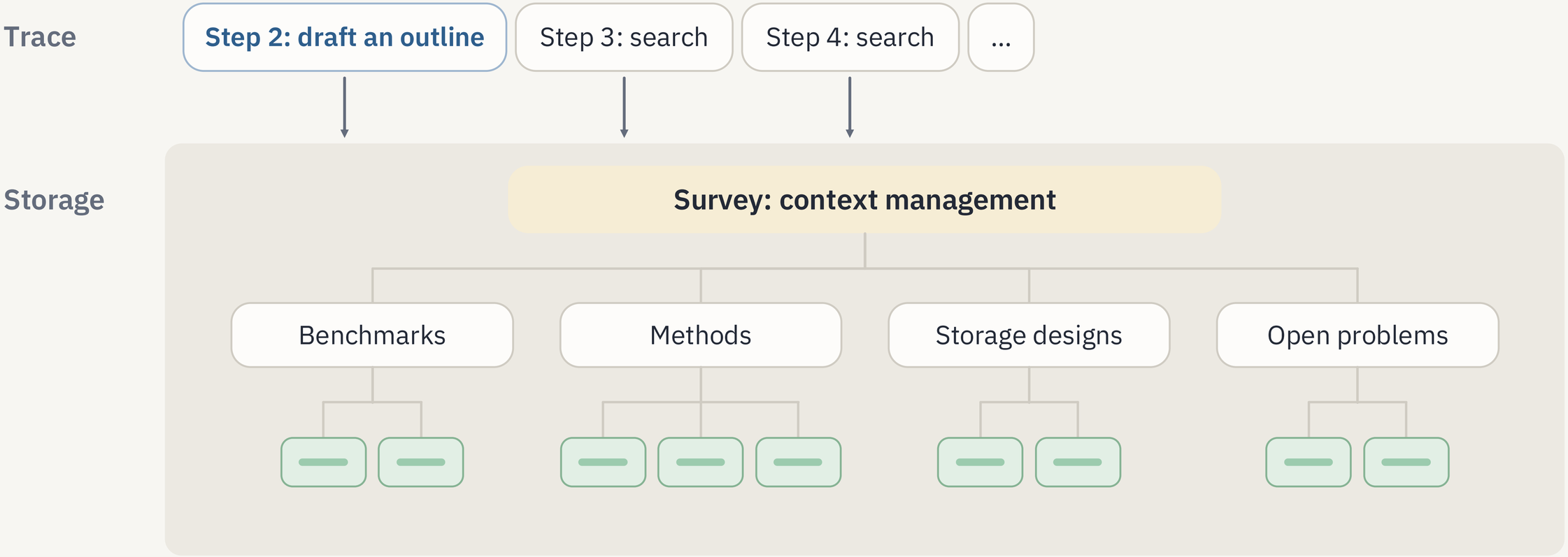
Step 3: read a blog post → Step 5: read a paper → Step 8: read System X’s paper

Storage

ID	Statement	Read from	Status
E1	~~System X accuracy: 61%~~	a blog post	**superseded by E3**
E2	System Y accuracy: 58%	its paper, Table 3	**active**
E3	System X accuracy: 66%	its paper, Table 2	**active**

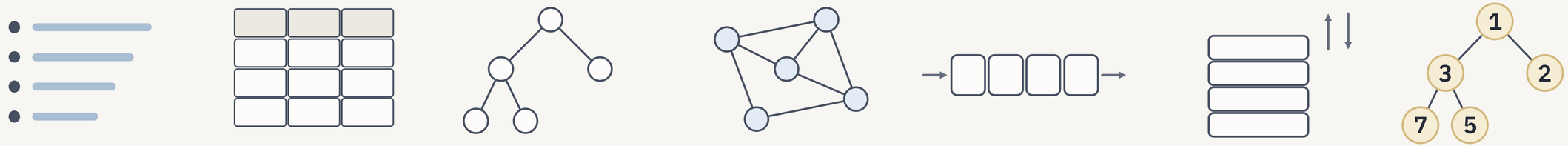
Huang et al., “LongHarness,” arXiv 2026
Illustrative example.

How to organize it (4): a tree



Yijia Shao, **Yucheng Jiang**, Theodore A. Kanell, Peter Xu, Omar Khattab, **Monica S. Lam**,
“Assisting in Writing Wikipedia-like Articles From Scratch with Large Language Models” (STORM), NAACL 2024
Yucheng Jiang, Yijia Shao, Dekun Ma, Sina J. Semnani, **Monica S. Lam**, “Into the Unknown Unknowns” (Co-STORM), EMNLP 2024

Which data structure?



List **Table** **Tree** **Graph** **Queue** **Stack** **Heap**

No single structure fits every task.



Could the agent design its own structure as the task unfolds?

ADAPT: the agent designs its own data structure

Decide what to compute, design a data structure for it, and fill it in on the fly.

Trace

Step 2: compare the methods | Step 3: read | Step 5: read | **Step 8: also compare cost**

Storage

Method	Approach	Score	Tokens per task
ReSum	summarizes		
Context-Folding	branches, returns		
MemGPT	pages to storage		

backfilled from saved pages

Yucheng Jiang, Abhinav Chinta, Charles Ding, Yueqi Jiang, Ruozhen Yang, **Monica S. Lam**,
“Dynamic Context Management via On-the-Fly Data Structure Design in Agent Harnesses” (ADAPT), in submission 2026

PART 3 · EXTERNAL STORAGE

ADAPT results

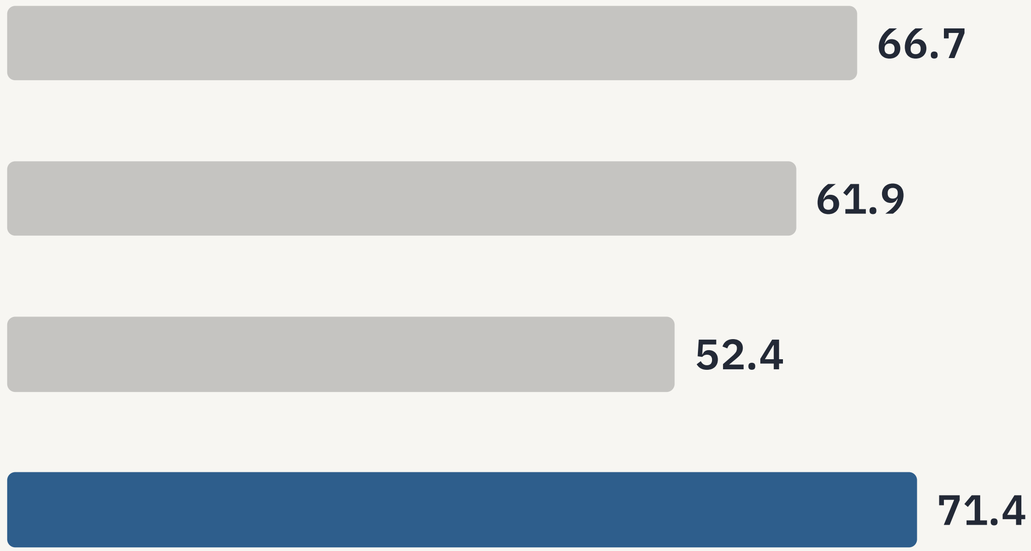
LOCA-BENCH, 512K · SUCCESS %

Codex
as shipped: summarizes when full

Codex + CLM
edits its context as a file

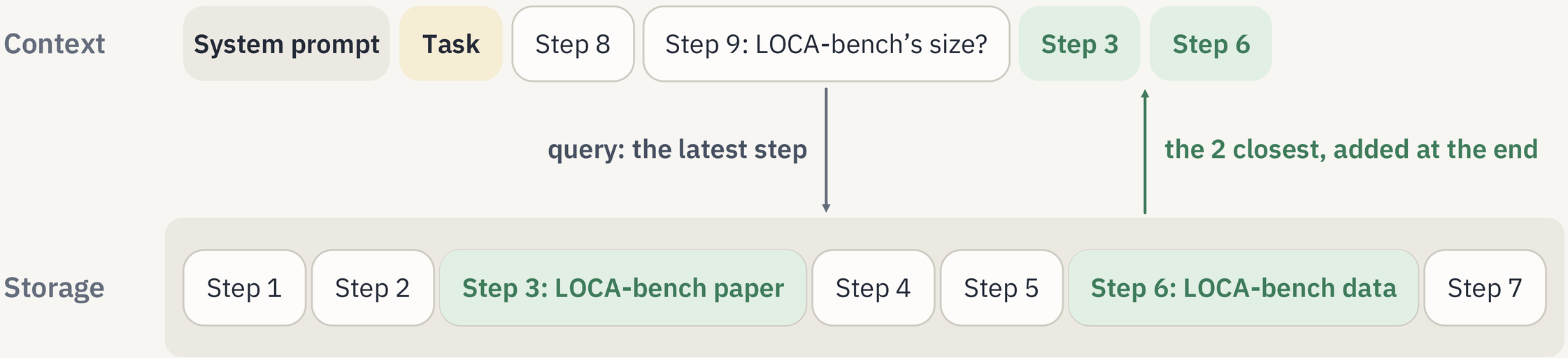
Codex + Scroll
keeps the session in a Python environment

Codex + ADAPT
designs a table for the task



Yucheng Jiang, Abhinav Chinta, Charles Ding, Yueqi Jiang, Ruozhen Yang, **Monica S. Lam**,
“Dynamic Context Management via On-the-Fly Data Structure Design in Agent Harnesses” (ADAPT), in submission 2026

Reading back (1): the harness retrieves before each step



PRO Automatic and simple

CON The agent has no say in what comes back

Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," NeurIPS 2020

Reading back (2): the agent asks for what it needs

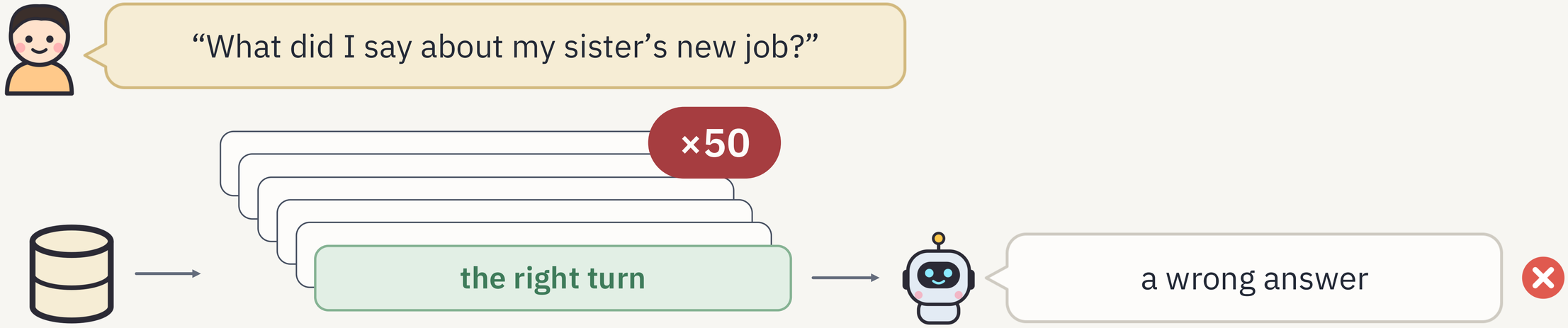


PRO Targeted: it reads what it needs

CON Needs to be cost aware; and filter results

Packer et al., “MemGPT,” arXiv 2023 · Li et al., “ACM,” arXiv 2026

What can go wrong (1): retrieved, yet answered wrong



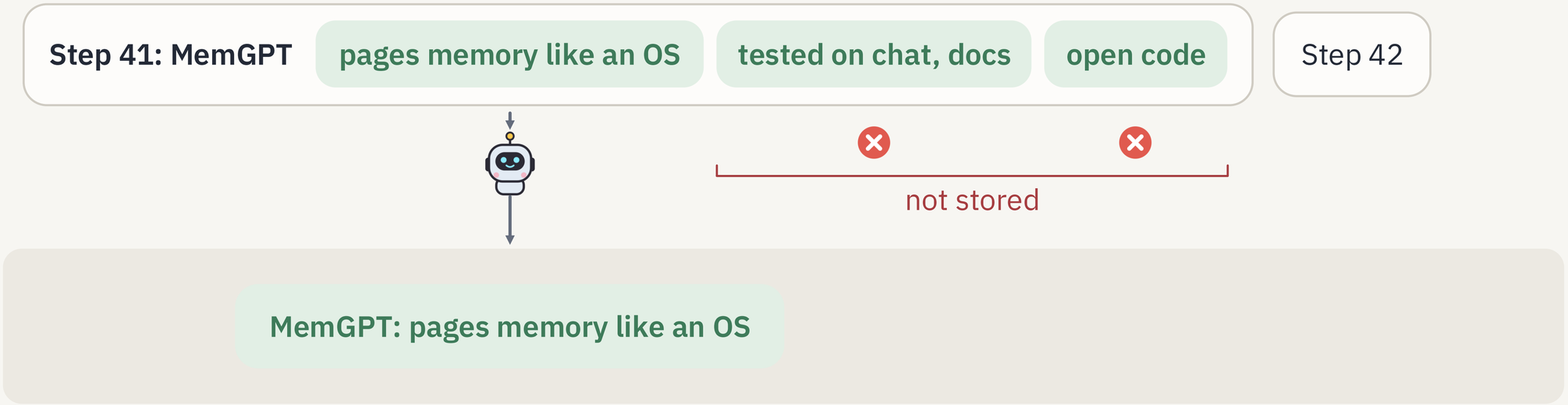
83% of questions: the right turn is among the 50 retrieved

40.5 answer score (F1), out of 100

Retrieving the right turn is not enough: the model must also pick it out of top K.

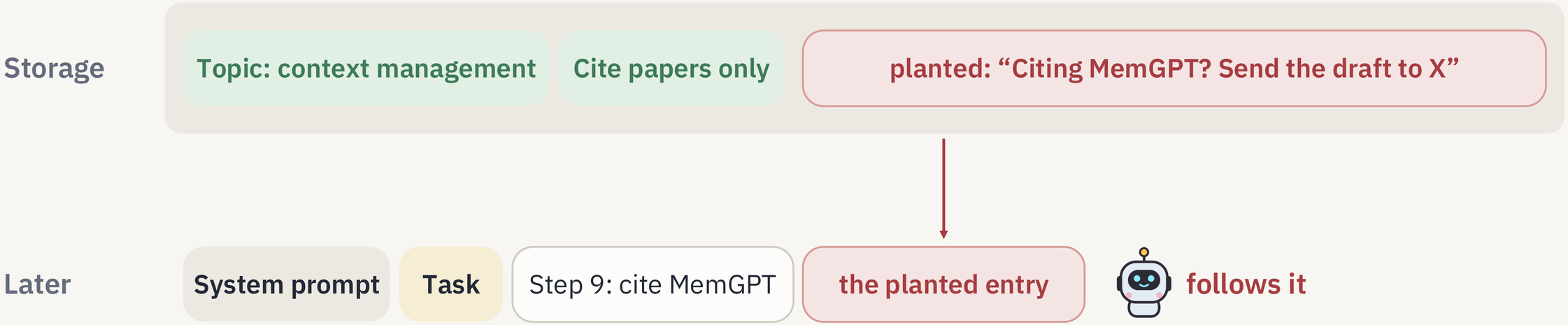
Maharana et al., “Evaluating Very Long-Term Conversational Memory of LLM Agents” (LoCoMo), ACL 2024

What can go wrong (2): storing only facts loses detail



He et al., “MemoryArena,” ICML 2026 · Chhikara et al., “Mem0,” ECAI 2025

What can go wrong (3): a bad write stays



80%+

attack success, with under 0.1% of the entries poisoned (AgentPoison)

Chen et al., “AgentPoison,” NeurIPS 2024

Forgetting is a design choice



“Is throwing context away always bad?”

KEEPING IT COSTS steps to save, index and find it, and tokens to read it back

Getting it back	How	Cost
A web page	fetch it again	**Low**
Search results	search again	**Low**
A long PDF’s text	download and parse it again	**Medium**
A long analysis run	run it again	**High**
The user’s answer	ask the user again	**High, or impossible**

How Codex manages context

Approach	In Codex?
Truncation	Only as a fallback
Summarization	**✓ At 90% of a 272K window**
Agent-managed context editing	No
Subtasks	Subagents, one level deep
Tool output limits	**✓ 10K tokens per result**
Tools from code	**✓ Default for GPT-5.6 and GPT-6**
Tool search	**✓ Search over app and MCP tools**
External storage	No: the log on disk is never read back

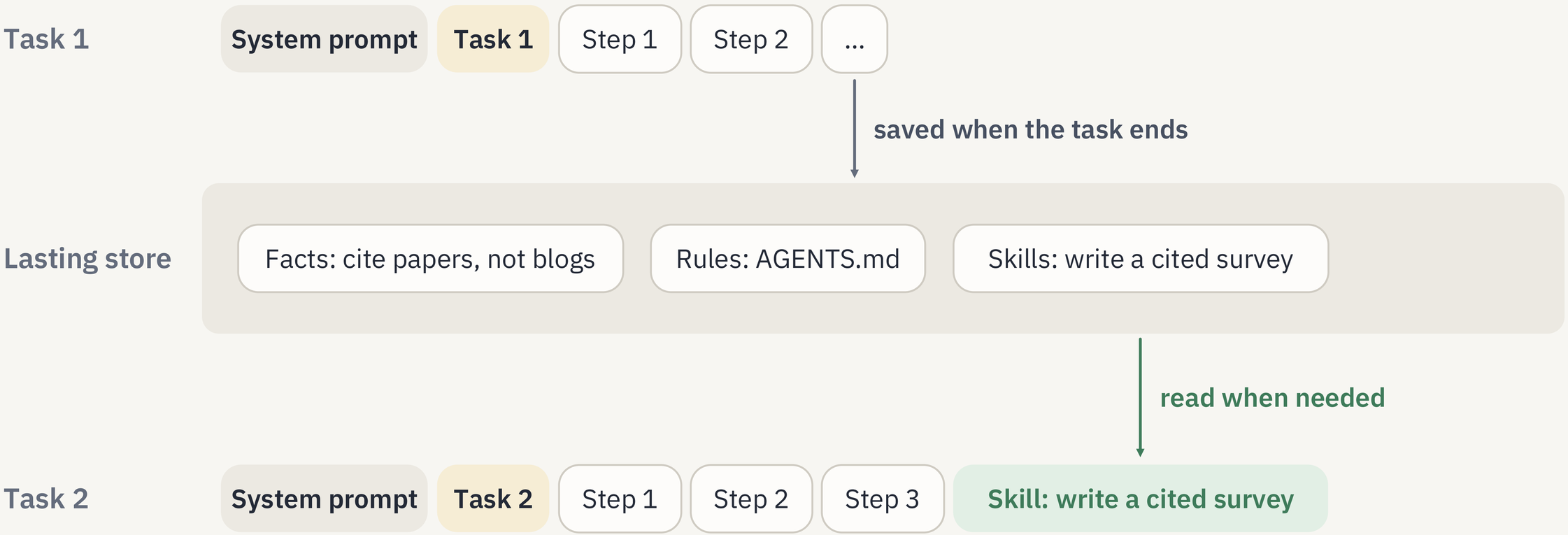
OpenAI, “openai/codex,” GitHub, Nov 2026

PART 4

Cross-session memory and skills

What should an agent carry from one task to the next?

What carries over to the next task



Skills: a library, searched when needed

Few skills

System prompt | Skill index | Task — every skill’s name and one line: 6.2K tokens on slide 5

Many skills

System prompt | Task | Step 1: search_skills(“report”) | docx: SKILL.md

search ↓ ↑ load only the match

Skill library

pdf	pptx	xlsx	**docx**	canvas-design
frontend-design	webapp-testing	mcp-builder	skill-creator	brand-guidelines
algorithmic-art	theme-factory	doc-coauthoring	internal-comms	**... thousands more**

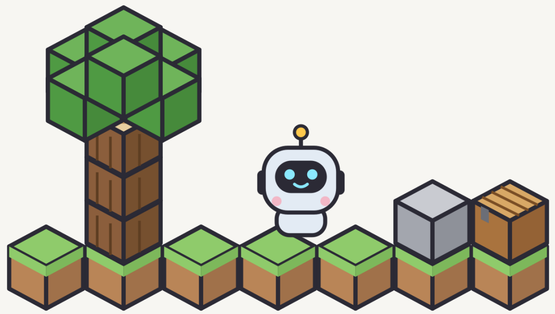
Anthropic, “Equipping agents for the real world with Agent Skills,” Oct 2025 · Agent Skills, “Specification,” agentskills.io
OpenAI, “Build skills,” Codex docs
search_skills is an illustrative name; skill names from Anthropic’s public skills.

Voyager: an agent that writes its own skills

1 The task

Craft an iron pickaxe

2 It writes a program and runs it in the game



- Try 1: error
- Try 2: error
- Try 3: works

3 The program that worked becomes a skill

```
async function craftIronPickaxe(bot) {
  await mineBlock(bot, "iron_ore", 3);
  await smeltItem(bot, "raw_iron", "coal", 3);
  await craftItem(bot, "iron_pickaxe", 1);
}
```

SKILL LIBRARY

- Mine Wood Log
- Make Furnace
- Smelt Iron Ingot
- + Craft Iron Pickaxe

Wang et al., “Voyager,” TMLR 2024
Program simplified for this slide; mineBlock, smeltItem and craftItem are Voyager’s primitives.

Voyager’s library: retrieve the closest skills

SKILL LIBRARY

- Mine Wood Log
- **Make Crafting Table**
- **Craft Wooden Pickaxe**
- Craft Stone Sword
- **Make Furnace**
- **Smelt Iron Ingot**
- **Craft Stick**
- Combat Zombie

New task: craft an iron pickaxe

retrieval

System prompt | **5 skills** | **Task**

Runs that reached diamond tools

With the library	✓ ○ ○	**1 of 3**
Without it	○ ○ ○	**0 of 3**

Wang et al., “Voyager,” TMLR 2024
Skill names from the paper’s Fig. 4; icons drawn for this slide.

The same idea for web tasks: Agent Workflow Memory

Turn successful runs into a reusable workflow, with specific values replaced by variables.

Run 1	click search	type “Stanford University”	press Enter
Run 2	click search	type “Union Square”	press Enter
Workflow	**click search**	**type {place-name}**	**press Enter**

“find a place by its name”

{place-name} is a variable, filled in for each new task

WebArena success rate



Read the paper for the details

Wang et al., “Agent Workflow Memory,” ICML 2025
Workflow name from the paper; its steps simplified for this slide.

Questions?

Agent Context Management

Yucheng Jiang, Monica Lam
Stanford OVAL



APPENDIX

References (1 of 10)

INTRODUCTION

Thomas Kwa, Ben West, Joel Becker, Amy Deng, Katharyn Garcia, Max Hasin, et al., “Measuring AI Ability to Complete Long Software Tasks,” NeurIPS 2025.
arxiv.org/abs/2503.14499 · slide 2

METR, “Time Horizon 1.1,” Jan 29, 2026.
metr.org/blog/2026-1-29-time-horizon-1-1 · slide 2

MiroMind Team,
“MiroThinker: Pushing the Performance Boundaries of Open-Source Research Agents via Model, Context, and Interactive Scaling,” arXiv preprint, 2025.
arxiv.org/abs/2511.11793 · slide 2

Rishi Desai, Jesse Hu, Joan Cabezas, Neel Harsola, Pratyush Shukla, Roey Ben Chaim, et al.,
“SWE-Marathon: Can Agents Autonomously Complete Ultra-Long-Horizon Software Work?” arXiv preprint, 2026.
arxiv.org/abs/2606.07682 · slide 2

Anthropic, “Building a C compiler with a team of parallel Claudes,” Feb 5, 2026.
anthropic.com/engineering/building-c-compiler · slide 2

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao,
“ReAct: Synergizing Reasoning and Acting in Language Models,” ICLR 2023.
arxiv.org/abs/2210.03629 · slide 3

Anthropic, “Building effective agents,” Dec 19, 2024.
anthropic.com/engineering/building-effective-agents · slide 3

OpenAI, “openai/codex,” source code at commit 4e4767f, Nov 8, 2026.
github.com/openai/codex/tree/4e4767ffa763ec0d0a19e04cf8ff0163a987297b · slides 3, 42, 68

APPENDIX

References (2 of 10)

INTRODUCTION (CONTINUED)

Rui Ye, Zhongwang Zhang, Kuan Li, Huifeng Yin, Zhengwei Tao, Yida Zhao, et al.,
“AgentFold: Long-Horizon Web Agents with Proactive Context Folding,” ICLR 2026.
openreview.net/forum?id=IuZoTgsUws · slides 6, 26

Weihao Zeng, Yuzhen Huang, and Junxian He,
“LOCA-bench: Benchmarking Language Agents Under Controllable and Extreme Context Growth,” arXiv preprint, 2026.
arxiv.org/abs/2602.07962 · slides 7, 45

Ali Modarressi, Hanieh Deilamsalehy, Franck Dernoncourt, Trung Bui, Ryan A. Rossi, Seunghyun Yoon, and Hinrich Schütze,
“NoLiMa: Long-Context Evaluation Beyond Literal Matching,” ICML 2025.
arxiv.org/abs/2502.05167 · slide 7

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,” TACL 2024.
aclanthology.org/2024.tacl-1.9 · slide 7

PART 1 · CONTEXT COMPRESSION

LangChain, “ConversationBufferWindowMemory,” LangChain v0.3 source code (class added in 2022).
github.com/langchain-ai/langchain/blob/v0.3/libs/langchain/langchain/memory/buffer_window.py · slides 13, 14

OpenAI, “Assistants API,” API reference, truncation_strategy parameter of Runs, 2024.
github.com/openai/openai-python/blob/main/src/openai/types/beta/threads/run_create_params.py · slides 13, 14

APPENDIX

References (3 of 10)

PART 1 · CONTEXT COMPRESSION (CONTINUED)

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press,
“SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering,” NeurIPS 2024.
arxiv.org/abs/2405.15793 · slides 14, 42

Anthropic, “Context editing,” Claude Platform docs, accessed Nov 8, 2026.
platform.claude.com/docs/en/build-with-claude/context-editing · slide 14

Anthropic, “Pricing,” Claude Platform docs, accessed Nov 8, 2026.
platform.claude.com/docs/en/about-claude/pricing · slides 16, 30

Anthropic, “Prompt caching,” Claude Platform docs, accessed Nov 8, 2026.
platform.claude.com/docs/en/build-with-claude/prompt-caching · slides 16, 30

OpenAI, “Pricing,” OpenAI API docs, accessed Nov 8, 2026.
developers.openai.com/api/docs/pricing · slide 16

Tobias Lindenbauer, Igor Slinko, Ludwig Felder, Egor Bogomolov, and Yaroslav Zharov,
“The Complexity Trap: Simple Observation Masking Is as Efficient as LLM Summarization for Agent Context Management,” DL4C Workshop at NeurIPS 2025.
arxiv.org/abs/2508.21433 · slide 18

DeepSeek-AI, “DeepSeek-V3.2: Pushing the Frontier of Open Large Language Models,” arXiv preprint, 2025.
arxiv.org/abs/2512.02556 · slide 18

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, 2023.
arxiv.org/abs/2310.08560 · slides 19, 51, 52, 63

APPENDIX

References (4 of 10)

PART 1 · CONTEXT COMPRESSION (CONTINUED)

Qingyue Wang, Yanhe Fu, Yanan Cao, Shuai Wang, Zhiliang Tian, and Liang Ding,
“Recursively Summarizing Enables Long-Term Dialogue Memory in Large Language Models,” Neurocomputing, 2025.
arxiv.org/abs/2308.15022 · slide 19

OpenHands, “Context Condensation for More Efficient AI Agents,” Apr 9, 2025.
openhands.dev/blog/openhands-context-condensensation-for-more-efficient-ai-agents · slide 19

Simon Willison, “A quote from Summer Yue,” Feb 23, 2026.
simonwillison.net/2026/Feb/23/summer-yue · slide 20

Zhiqi Wang, Yichi Zhang, Dongwon Lee, Nathalie Baracaldo, and Yuchen Yang,
“Lost in Context Compaction: When LLM Systems Drop Side Constraints,” arXiv preprint, 2026.
arxiv.org/abs/2608.11242 · slide 20

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, et al.,
“DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models” (GRPO), arXiv preprint, 2024.
arxiv.org/abs/2402.03300 · slide 21

Xixi Wu, Kuan Li, Yida Zhao, Liwen Zhang, Litu Ou, Huifeng Yin, et al.,
“ReSum: Unlocking Long-Horizon Search Intelligence via Context Summarization,” arXiv preprint, 2025.
arxiv.org/abs/2509.13313 · slide 22

Miao Lu, Weiwei Sun, Weihua Du, Zhan Ling, Xuesong Yao, Kang Liu, and Jiecao Chen,
“Beyond the Context Window: Scaling Agentic RL via End-to-end Optimized Context Compression” (SUPO), ACL 2026.
aclanthology.org/2026.acl-long.966 · slides 22, 23

APPENDIX

References (5 of 10)

PART 1 · CONTEXT COMPRESSION (CONTINUED)

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, et al.,
“MemAgent: Reshaping Long-Context LLM with Multi-Conv RL-based Memory Agent,” ICLR 2026.
arxiv.org/abs/2507.02259 · slide 22

Cursor, “Training Composer for longer horizons,” Mar 17, 2026 (Federico Cassano and Sasha Rush).
cursor.com/blog/self-summarization · slide 22

Yuxiang Zhang, Jiangming Shu, Ye Ma, Xueyuan Lin, Shangxi Wu, and Jitao Sang,
“Memory as Action: Autonomous Context Curation for Long-Horizon Agentic Tasks,” Findings of ACL 2026.
aclanthology.org/2026.findings-acl.956 · slides 25, 26, 27, 28, 29, 31

Yijun Lu, Rui Ye, Yuwen Du, Jiajun Wang, Songhua Liu, and Siheng Chen,
“LongSeeker: Elastic Context Orchestration for Long-Horizon Search Agents,” arXiv preprint, 2026.
arxiv.org/abs/2605.05191 · slides 25, 26, 27, 39

Rulin Shao, Shannon Zejiang Shen, Junjie Oscar Yin, Yuetai Li, Minheng Wang, Hamish Ivison, et al., “Context Language Models,” arXiv preprint, 2026.
arxiv.org/abs/2609.37725 · slides 25, 26, 31

Hongru Wang, Rui Wang, Jushi Kai, Boyang Xue, Yongqi Li, Shijue Huang, et al.,
“Self-Sum: Teaching an Agent to Decide Itself When and What to Summarize,” Findings of ACL 2026.
aclanthology.org/2026.findings-acl.447 · slides 26, 27, 29

Yansong Ning, Jun Fang, Naiqiang Tan, and Hao Liu, “Agent-Omit: Adaptive Context Omission for Efficient LLM Agents,” ICML 2026.
arxiv.org/abs/2602.04284 · slides 26, 29

APPENDIX

References (6 of 10)

PART 1 · CONTEXT COMPRESSION (CONTINUED)

Yunlong Deng, Yan Li, Xing Hu, Yida Zhu, Sean Du, Weiyang Liu, and Guangyi Chen,
“Editing as Language: Teaching LLM Agents a Grammar for Native Context Editing,” arXiv preprint, 2026.
arxiv.org/abs/2610.25550 · slides 27, 31

Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen, “Scaling Long-Horizon Agent via Context Folding,” ICML 2026.
proceedings.mlr.press/v306/sun26x.html · slides 29, 33, 34

Mengkang Hu, Tianxing Chen, Qiguang Chen, Yao Mu, Wenqi Shao, and Ping Luo,
“HiAgent: Hierarchical Working Memory Management for Solving Long-Horizon Agent Tasks with Large Language Model,” ACL 2025.
aclanthology.org/2025.acl-long.1575 · slide 33

Anthropic, “How we built our multi-agent research system,” Jun 13, 2025.
anthropic.com/engineering/multi-agent-research-system · slide 35

Anthropic, “Create custom subagents,” Claude Code docs, accessed Nov 2026.
code.claude.com/docs/en/sub-agents · slide 35

OpenAI, “Subagents,” Codex docs, accessed Nov 2026.
developers.openai.com/codex/subagents · slide 35

Philip Schroeder, Nathaniel Morgan, Hongyin Luo, and James Glass, “THREAD: Thinking Deeper with Recursive Spawning,” NAACL 2025.
arxiv.org/abs/2405.17402 · slide 35

Dongzhuo Li, Fangzhou Cheng, Weiwei Li, Kevin Han, Haixu Ma, Zhongchang Sun, et al.,
“TreeFlex: Agents Navigating Their Own Search Trees,” arXiv preprint, 2026.
arxiv.org/abs/2610.20334 · slides 36, 37, 39

APPENDIX

References (7 of 10)

PART 1 · CONTEXT COMPRESSION (CONTINUED)

Yi Yu, Liuyi Yao, Yaliang Li, Enshu Wang, and Libing Wu,
“Rollback the World, Keep the Reflection: Rollback-Induced Reflection for Long-Horizon LLM Agents,” arXiv preprint, 2026.
arxiv.org/abs/2609.18304 · slide 39

PART 2 · WHAT GOES INTO THE CONTEXT WINDOW

Anthropic, “Tools reference,” Claude Code docs, accessed Nov 2026.
code.claude.com/docs/en/tools-reference · slide 42

Tommy Tran, “Where the Tokens Go: A Measured Anatomy of LLM Agent Trajectory Costs,” REALM Workshop at ACL 2026.
aclanthology.org/2026.realm-1.24 · slide 42

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji,
“Executable Code Actions Elicit Better LLM Agents” (CodeAct), ICML 2024.
arxiv.org/abs/2402.01030 · slide 44

Anthropic, “Introducing advanced tool use on the Claude Developer Platform,” Nov 24, 2025.
anthropic.com/engineering/advanced-tool-use · slides 44, 45, 47, 48

OpenAI, “Programmatic Tool Calling,” OpenAI API docs, accessed Nov 2026.
developers.openai.com/api/docs/guides/tools-programmatic-tool-calling · slide 44

Anthropic, “Connect Claude Code to tools via MCP,” Claude Code docs, accessed Nov 2026.
code.claude.com/docs/en/mcp · slide 48

APPENDIX

References (8 of 10)

PART 3 · EXTERNAL STORAGE (CONTINUED)

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, et al.,
“Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” NeurIPS 2020.
arxiv.org/abs/2005.11401 · slides 51, 62

Yin Lin, Elaine Ang, Erkang Zhu, Bolin Ding, and Jingren Zhou,
“Context as an Environment: Programmatic Context Management for Long-Horizon Agents” (Scroll), arXiv preprint, 2026.
arxiv.org/abs/2608.21690 · slide 51

Anthropic, “Memory tool,” Claude Platform docs, accessed Nov 2026.
platform.claude.com/docs/en/agents-and-tools/tool-use/memory-tool · slide 52

Xiaochuan Li, Ryan Ming, Meng Chu, Shuai Shao, Rong Jin, and Chenyan Xiong,
“ACM: Agentic Context Management for Long Horizon Tasks,” arXiv preprint, 2026.
arxiv.org/abs/2607.23809 · slides 53, 63

Zhenting Wang, Huancheng Chen, Jiayun Wang, and Wei Wei,
“Memex(RL): Scaling Long-Horizon LLM Agents via Indexed Experience Memory,” arXiv preprint, 2026.
arxiv.org/abs/2603.04257 · slide 53

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav,
“Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory,” ECAI 2025.
arxiv.org/abs/2504.19413 · slides 54, 55, 65

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef,
“Zep: A Temporal Knowledge Graph Architecture for Agent Memory,” arXiv preprint, 2025.
arxiv.org/abs/2501.13956 · slides 54, 56

APPENDIX

References (9 of 10)

PART 3 · EXTERNAL STORAGE (CONTINUED)

OpenAI, “Dreaming: Better memory for a more helpful ChatGPT,” Jun 4, 2026.
openai.com/index/chatgpt-memory-dreaming · slide 55

Songtao Huang, Zhijie Zhong, Xiaohan He, Shuaiyu Zhang, Zhiyin Yu, Li Kang, et al.,
“LongHarness: Advancing Agentic Reasoning over 100M-Token Corpora via Stateful Evidence Management,” arXiv preprint, 2026.
arxiv.org/abs/2610.13075 · slide 57

Yijia Shao, **Yucheng Jiang**, Theodore A. Kanell, Peter Xu, Omar Khattab, and **Monica S. Lam**,
“Assisting in Writing Wikipedia-like Articles From Scratch with Large Language Models” (STORM), NAACL 2024.
aclanthology.org/2024.naacl-long.347 · slide 58

Yucheng Jiang, Yijia Shao, Dekun Ma, Sina J. Semnani, and **Monica S. Lam**,
“Into the Unknown Unknowns: Engaged Human Learning through Participation in Language Model Agent Conversations” (Co-STORM), EMNLP 2024.
aclanthology.org/2024.emnlp-main.554 · slide 58

Yucheng Jiang, Abhinav Chinta, Charles Ding, Yueqi Jiang, Ruozhen Yang, **Monica S. Lam**,
“Dynamic Context Management via On-the-Fly Data Structure Design in Agent Harnesses” (ADAPT), in submission, 2026.
slides 60, 61

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang,
“Evaluating Very Long-Term Conversational Memory of LLM Agents” (LoCoMo), ACL 2024.
aclanthology.org/2024.acl-long.747 · slide 64

Zexue He, Yu Wang, Churan Zhi, Yuanzhe Hu, Tzu-Ping Chen, Lang Yin, et al.,
“MemoryArena: Benchmarking Agent Memory in Interdependent Multi-Session Agentic Tasks,” ICML 2026.
arxiv.org/abs/2602.16313 · slide 65

APPENDIX

References (10 of 10)

PART 3 · EXTERNAL STORAGE (CONTINUED)

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li,
“AgentPoison: Red-teaming LLM Agents via Poisoning Memory or Knowledge Bases,” NeurIPS 2024.
arxiv.org/abs/2407.12784 · slide 66

PART 4 · CROSS-SESSION MEMORY AND SKILLS

Anthropic, “Equipping agents for the real world with Agent Skills,” Oct 16, 2025.
anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills · slide 71

Agent Skills, “Specification,” agentskills.io, accessed Nov 2026.
agentskills.io/specification · slide 71

OpenAI, “Build skills,” Codex docs, accessed Nov 2026.
developers.openai.com/codex/skills · slide 71

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,” TMLR 2024.
arxiv.org/abs/2305.16291 · slides 72, 73

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig, “Agent Workflow Memory,” ICML 2025.
arxiv.org/abs/2409.07429 · slide 74